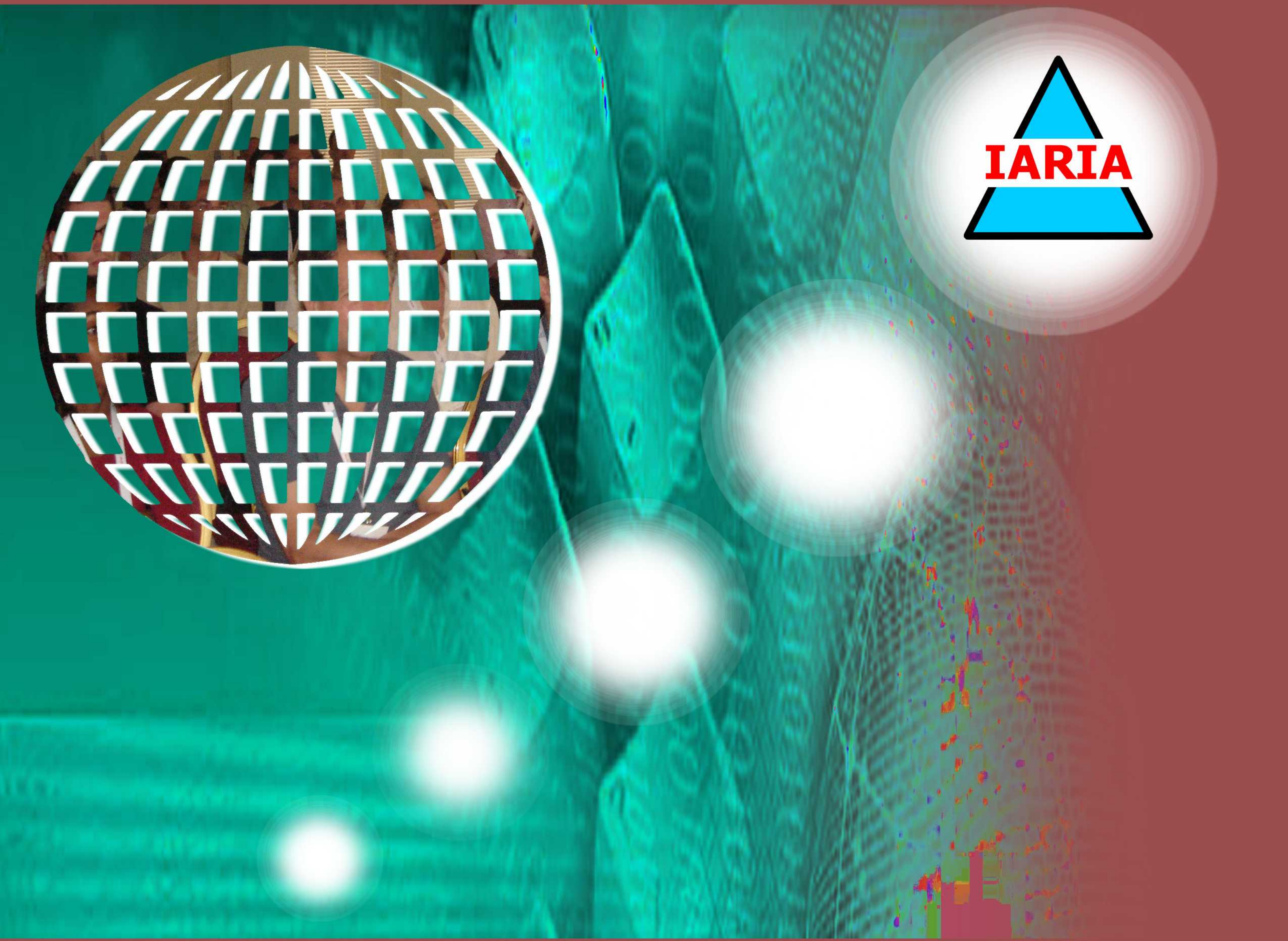
International Journal on Advances in Software



2018 vol. 11 nr. 1&2

The *International Journal on Advances in Software* is published by IARIA.
ISSN: 1942-2628
journals site: http://www.iariajournals.org
contact: petre@iaria.org

Sponsored by IARIA
www.iaria.org

International Journal on Advances in Software
Volume 11, Number 1 & 2, 2018

Andreas S. Andreou, Cyprus University of Technology - Limassol, Cyprus
Annalisa Appice, Università degli Studi di Bari Aldo Moro, Italy
Philip Azariadis, University of the Aegean, Greece
Thierry Badard, Université Laval, Canada
Muneera Bano, International Islamic University - Islamabad, Pakistan
Fabian Barbato, Technology University ORT, Montevideo, Uruguay
Peter Baumann, Jacobs University Bremen / Rasdaman GmbH Bremen, Germany
Gabriele Bavota, University of Salerno, Italy
Grigorios N. Beligiannis, University of Western Greece, Greece
Noureddine Belkhatir, University of Grenoble, France
Jorge Bernardino, ISEC - Institute Polytechnic of Coimbra, Portugal
Rudolf Berrendorf, Bonn-Rhein-Sieg University of Applied Sciences - Sankt Augustin, Germany
Ateet Bhalla, Independent Consultant, India
Fernando Boronat Seguí, Universidad Politecnica de Valencia, Spain
Pierre Borne, Ecole Centrale de Lille, France
Farid Bourennani, University of Ontario Institute of Technology (UOIT), Canada
Narhimene Boustia, Saad Dahlab University - Blida, Algeria
Hongyu Pei Breivold, ABB Corporate Research, Sweden
Carsten Brockmann, Universität Potsdam, Germany
Antonio Bucchiarone, Fondazione Bruno Kessler, Italy
Georg Buchgeher, Software Competence Center Hagenberg GmbH, Austria
Dumitru Burdescu, University of Craiova, Romania
Martine Cadot, University of Nancy / LORIA, France
Isabel Candal-Vicente, Universidad del Este, Puerto Rico
Juan-Vicente Capella-Hernández, Universitat Politècnica de València, Spain
Jose Carlos Metrolho, Polytechnic Institute of Castelo Branco, Portugal
Alain Casali, Aix-Marseille University, France
Yaser Chaaban, Leibniz University of Hanover, Germany
Savvas A. Chatzichristofis, Democritus University of Thrace, Greece
Antonin Chazalet, Orange, France
Jiann-Liang Chen, National Dong Hwa University, China
Shiping Chen, CSIRO ICT Centre, Australia
Wen-Shiung Chen, National Chi Nan University, Taiwan
Zhe Chen, College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, China PR
Po-Hsun Cheng, National Kaohsiung Normal University, Taiwan
Yoonsik Cheon, The University of Texas at El Paso, USA
Lau Cheuk Lung, INE/UFSC, Brazil
Robert Chew, Lien Centre for Social Innovation, Singapore
Andrew Connor, Auckland University of Technology, New Zealand
Rebeca Cortázar, University of Deusto, Spain
Noël Crespi, Institut Telecom, Telecom SudParis, France
Carlos E. Cuesta, Rey Juan Carlos University, Spain
Duilio Curcio, University of Calabria, Italy
Mirela Danubianu, "Stefan cel Mare" University of Suceava, Romania
Paulo Asterio de Castro Guerra, Tapijara Programação de Sistemas Ltda. - Lambari, Brazil

Cláudio de Souza Baptista, University of Campina Grande, Brazil
Maria del Pilar Angeles, Universidad Nacional Autonónoma de México, México
Rafael del Vado Vírseda, Universidad Complutense de Madrid, Spain
Giovanni Denaro, University of Milano-Bicocca, Italy
Nirmit Desai, IBM Research, India
Vincenzo Deufemia, Università di Salerno, Italy
Leandro Dias da Silva, Universidade Federal de Alagoas, Brazil
Javier Diaz, Rutgers University, USA
Nicholas John Dingle, University of Manchester, UK
Roland Dodd, CQUniversity, Australia
Aijuan Dong, Hood College, USA
Suzana Dragicevic, Simon Fraser University- Burnaby, Canada
Cédric du Mouza, CNAM, France
Ann Dunkin, Palo Alto Unified School District, USA
Jana Dvorakova, Comenius University, Slovakia
Hans-Dieter Ehrich, Technische Universität Braunschweig, Germany
Jorge Ejarque, Barcelona Supercomputing Center, Spain
Atilla Elçi, Aksaray University, Turkey
Khaled El-Fakih, American University of Sharjah, UAE
Gledson Elias, Federal University of Paraíba, Brazil
Sameh Elnikety, Microsoft Research, USA
Fausto Fasano, University of Molise, Italy
Michael Felderer, University of Innsbruck, Austria
João M. Fernandes, Universidade de Minho, Portugal
Luis Fernandez-Sanz, University of de Alcala, Spain
Felipe Ferraz, C.E.S.A.R, Brazil
Adina Magda Florea, University "Politehnica" of Bucharest, Romania
Wolfgang Fohl, Hamburg Universiy, Germany
Simon Fong, University of Macau, Macau SAR
Gianluca Franchino, Scuola Superiore Sant'Anna, Pisa, Italy
Naoki Fukuta, Shizuoka University, Japan
Martin Gaedke, Chemnitz University of Technology, Germany
Félix J. García Clemente, University of Murcia, Spain
José García-Fanjul, University of Oviedo, Spain
Felipe Garcia-Sanchez, Universidad Politecnica de Cartagena (UPCT), Spain
Michael Gebhart, Gebhart Quality Analysis (QA) 82, Germany
Tejas R. Gandhi, Virtua Health-Marlton, USA
Andrea Giachetti, Università degli Studi di Verona, Italy
Afzal Godil, National Institute of Standards and Technology, USA
Luis Gomes, Universidade Nova Lisboa, Portugal
Diego Gonzalez Aguilera, University of Salamanca - Avila, Spain
Pascual Gonzalez, University of Castilla-La Mancha, Spain
Björn Gottfried, University of Bremen, Germany
Victor Govindaswamy, Texas A&M University, USA
Gregor Grambow, AristaFlow GmbH, Germany
Carlos Granell, European Commission / Joint Research Centre, Italy

Clement Leung, Hong Kong Baptist University, Hong Kong
Weidong Li, University of Connecticut, USA
Corrado Loglisci, University of Bari, Italy
Francesco Longo, University of Calabria, Italy
Sérgio F. Lopes, University of Minho, Portugal
Pericles Loucopoulos, Loughborough University, UK
Alen Lovrencic, University of Zagreb, Croatia
Qifeng Lu, MacroSys, LLC, USA
Xun Luo, Qualcomm Inc., USA
Stephane Maag, Telecom SudParis, France
Ricardo J. Machado, University of Minho, Portugal
Maryam Tayefeh Mahmoudi, Research Institute for ICT, Iran
Nicos Malevris, Athens University of Economics and Business, Greece
Herwig Mannaert, University of Antwerp, Belgium
José Manuel Molina López, Universidad Carlos III de Madrid, Spain
Francesco Marcelloni, University of Pisa, Italy
Eda Marchetti, Consiglio Nazionale delle Ricerche (CNR), Italy
Gerasimos Marketos, University of Piraeus, Greece
Abel Marrero, Bombardier Transportation, Germany
Adriana Martin, Universidad Nacional de la Patagonia Austral / Universidad Nacional del Comahue, Argentina
Goran Martinovic, J.J. Strossmayer University of Osijek, Croatia
Paulo Martins, University of Trás-os-Montes e Alto Douro (UTAD), Portugal
Stephan Mäs, Technical University of Dresden, Germany
Constandinos Mavromoustakis, University of Nicosia, Cyprus
Jose Merseguer, Universidad de Zaragoza, Spain
Seyedeh Leili Mirtaheri, Iran University of Science & Technology, Iran
Lars Moench, University of Hagen, Germany
Yasuhiko Morimoto, Hiroshima University, Japan
Antonio Navarro Martín, Universidad Complutense de Madrid, Spain
Filippo Neri, University of Naples, Italy
Muaz A. Niazi, Bahria University, Islamabad, Pakistan
Natalja Nikitina, KTH Royal Institute of Technology, Sweden
Roy Oberhauser, Aalen University, Germany
Pablo Oliveira Antonino, Fraunhofer IESE, Germany
Rocco Oliveto, University of Molise, Italy
Sascha Opletal, Universität Stuttgart, Germany
Flavio Oquendo, European University of Brittany/IRISA-UBS, France
Claus Pahl, Dublin City University, Ireland
Marcos Palacios, University of Oviedo, Spain
Constantin Paleologu, University Politehnica of Bucharest, Romania
Kai Pan, UNC Charlotte, USA
Yiannis Papadopoulos, University of Hull, UK
Andreas Papasalouros, University of the Aegean, Greece
Rodrigo Paredes, Universidad de Talca, Chile
Päivi Parviainen, VTT Technical Research Centre, Finland
João Pascoal Faria, Faculty of Engineering of University of Porto / INESC TEC, Portugal

Ergin Tari, Istanbul Technical University, Turkey
Steffen Thiel, Furtwangen University of Applied Sciences, Germany
Jean-Claude Thill, Univ. of North Carolina at Charlotte, USA
Pierre Tiako, Langston University, USA
Božo Tomas, HT Mostar, Bosnia and Herzegovina
Davide Tosi, Università degli Studi dell'Insubria, Italy
Guglielmo Trentin, National Research Council, Italy
Dragos Truscan, Åbo Akademi University, Finland
Chrisa Tsinaraki, Technical University of Crete, Greece
Roland Ukor, FirstLinq Limited, UK
Torsten Ullrich, Fraunhofer Austria Research GmbH, Austria
José Valente de Oliveira, Universidade do Algarve, Portugal
Dieter Van Nuffel, University of Antwerp, Belgium
Shirshu Varma, Indian Institute of Information Technology, Allahabad, India
Konstantina Vassilopoulou, Harokopio University of Athens, Greece
Miroslav Velev, Aries Design Automation, USA
Tanja E. J. Vos, Universidad Politécnica de Valencia, Spain
Krzysztof Walczak, Poznan University of Economics, Poland
Yandong Wang, Wuhan University, China
Rainer Weinreich, Johannes Kepler University Linz, Austria
Stefan Wesarg, Fraunhofer IGD, Germany
Wojciech Wiza, Poznan University of Economics, Poland
Martin Wojtczyk, Technische Universität München, Germany
Hao Wu, School of Information Science and Engineering, Yunnan University, China
Mudasser F. Wyne, National University, USA
Zhengchuan Xu, Fudan University, P.R.China
Yiping Yao, National University of Defense Technology, Changsha, Hunan, China
Stoyan Yordanov Garbatov, Instituto de Engenharia de Sistemas e Computadores - Investigação e Desenvolvimento, INESC-ID, Portugal
Weihai Yu, University of Tromsø, Norway
Wenbing Zhao, Cleveland State University, USA
Hong Zhu, Oxford Brookes University, UK
Qiang Zhu, The University of Michigan - Dearborn, USA

International Journal on Advances in Software
Volume 11, Numbers 1 & 2, 2018

CONTENTS

THE DEVELOPMENT OF RELIABLE ESTIMATION METHODS FOR THE USAGE OF TEST AUTOMATION IN AN AGILE ENVIRONMENT

Short Paper

Jos van Rooyen
Identify Test Services BV
Rosmalen, the Netherlands
jos.van.rooyen@identify.nl

***Abstract*- One way to ensure the quality of software is by testing it. Running tests can be done both manually and automatically. Test automation, with its ups and downs, has been the centre of attention for many years. It is usually underestimated what implementing test automation entails and the impact it has on an organization, especially the estimation of the required effort. Adding test automation within the entire range of testing measures requires extra human capacity, both for the initial set up and the maintenance of the automatized tests apart from test implementation. The question is how much human capacity is needed in order to test automatize the functionality which has to be tested automatically? This article describes several methods to estimate the required effort for test automation and the approach to collect the required data. By applying the described estimation methods via a case study key figures can be defined to estimate the required effort for test automation projects.**

Keywords-Estimation; Test Automation; Agile; Testing; Return on Investment; Future proof.

I. INTRODUCTION

During Valid2016 the first ideas were presented [1] how to estimate test automation in an Agile environment. Based on the discussion during the conference and further elaboration of the topic, this paper describes several methods to estimate the required effort for test automation. Inside testing [2], test automation is one way to ensure the quality of software.

What is meant by testing software is the following [3]:

“The process consisting of all lifecycle activities, both static and dynamic, concerned with planning, preparation and evaluation of software products and related work products to determine that they satisfy specified requirements, to demonstrate that they fit for purpose and to detect defects.”

Running tests can be done both manually and automatically. Test automation, with its ups and downs, has been the centre of attention for many years. It is usually underestimated what implementing test automation entails and the impact it has on an organization [4][5]. Because of the rising popularity of Agile [6] and the implementation of continuous deployment and development [7], it seems that test automation is taking on a fixed position. The most important reason for this is that the amount of work is no longer manageable to be done manually [8].

Adding test automation within the entire range of testing measures requires extra human capacity, both for the initial set up and the maintenance of the automatized tests apart from test implementation. The question is how much human capacity is needed in order to test automatize the functionality which has to be tested automatically?

Test budgeting has been a problem since the beginning [9]. Several methods have been developed [10][11][12], but they do not always produce the correct results. Practice shows that significantly more time is needed than was budgeted at the start of the project.

The question is how to get a grip on this in order to make reliable predictions concerning the necessary capacity. Based on previous methods [10][11][12], which are the utilized ways of budgeting within the Agile methodology [13], three ways of thinking have been developed to budget automatized test capacity. These three ways of thinking are described in this article. However, they still have to be tested in practice.

The fundamental principle in this article is a structural, future proof design of test automation within the Agile developed methodology. That means the following: designing test automation in such a way that developed automatized tests cannot only be executed, reused, and easily transferred to others today, but also in the future, and done in such a way that maintenance effort is minimal.

The paper has the following structure. Section II describes the causes of poor test budgets on behalf of test automation. Section III describes the general elements that affect the required test capacity. Section IV will give insight into budgeting future proof elements on behalf of test automation. Section V discusses the three ways of budgeting. A detailed example has been included in Section VI. Section VII describes the collection of the data and the approach to classify into the described estimation methods. Return on investment is dealt with in Section VIII. Lastly, Section IX describes the conclusions and future work.

II. Causes of Poor Test Automation Budgeting

As indicated in the introduction, budgeting within ICT is a common problem [9]. Which causes are at the core of this? A couple of reasons can be found.

Using new development and or programming techniques of which there is not enough knowledge. Not questioning the desired functionality enough which causes new problems to arise during the implementation and test phase. Unfamiliarity with the quality of the software in the beginning is another reason. Furthermore, the quality of the persons involved, such as the tester or developer, plays a role. Is someone sufficiently skilled to make a solid budget? [12][11][10].

What we see in practice is that experience numbers are hardly recorded, if recorded at all. This is especially the case for budgeting test automation. A short research in the ISBSG database [14] shows that only three projects have been recorded in which Agile development technique is combined with automatized testing. Of only 1 project out of these three, the delivered test effort has been registered (see Table I).

Table I: Analysis ISBSG-database for the attention of test projects

#projects	Way of testing		Agile development methodology		Test effort known	
	Manual	Automatized	Y	N	Y	N
95	29	66	3	63	1	2

It becomes clear from this analysis that there is no useful data available to draw conclusions.

In order to try to answer the question: "How is budgeting done in an Agile development environment concerning test automation," a survey has been conducted in which 100 people participated. The results are recorded in Table II.

Table II: Results survey way of budgeting test automation

Way of budgeting	Number
Not budgeted	2
Percentage available time	1
Pokering of the effort	3
Experience based	5
No response	89
Total	100

As Table II shows, no reliable results can be extracted from the survey. Despite a reminder, response was very low. Both the results of the survey and the analysis of the ISBSG database, which show comparable results, were a trigger to keep on thinking of ways how to budget test automation reliably and predictably.

A first draft was made during the Valid2016 conference where the first ideas were drafted during the presentation: "Estimation of test automation in an Agile Environment" in which the following question was discussed: "How to estimate the required effort in an Agile environment regarding test automation" [1].

During this presentation three ways of thinking were sketched how test automation can be budgeted in an Agile development environment in order to set up test automation in a structured and future proof manner. This article elaborates on the presentation whereby received input has been included in further working out the ways of thinking. Besides the manner of budgeting, each approach has a number of general elements which influence the eventual budget for test automation. These general elements will be elaborated on first.

III. General Elements Influencing Budgeting of Test Automation

Apart from the required budget to automatize the test scripts, there are several preconditional elements which influence the required budget for test automation. No matter at which level in the organization (project, division or company level) [15] you wish to implement test automation, you will have to deal with these elements. Dependent on the level at which you would like to implement test automation, the impact on the organization will be bigger. If you focus test automation on company level instead of a project or individual sprint, the involved elements will have a wider impact. The elements can be separated in the so-called initial costs and continuity costs. The initial costs are those which you have when setting up and developing test automation for the first time in an organization.

Continuity costs are costs which have to be made after the introduction of test automation in order to maintain and expand (if necessary) test automation. In Table III, the relevant elements are mentioned with an indication how these elements can be measured and a short explanation.

Table III: Initial and continuity elements test automation

Component	Element	Unit of measure	Explanation
Initial costs	People	#people Costs per day	Number of people that are going to work on test automation
	Test tools	#tools Price per license	Type and number of test tools to purchase aligned with various development platforms
	Installation costs	#days Costs per day	Installation of test tools in the ICT- landscape
	Test data	#days Costs per day	Choosing a working method: formulate test data requirements, making synthetic test data, scrambling production data to use as test data. Taking privacy into account [16]
	Education	# days	Number of required educations/courses
	Frequency of usage	#runs	How often is test automation used?
	Virtualization		Is virtualization used?

	Number of integrations with surrounding systems	#integrations costs per integration	Which integrations are relevant and how are they mutually dependent?
	Support which company objectives	n/a (not applicable)	Which strategic objectives have to be supported?
Continuity	License costs test tools	Price per license	Annual costs on behalf of the test tools
	Maintenance test scripts	Modification frequency	Percentage time reserved for maintenance of the test scripts
	Additional training	#days #days upgrade	Required training for new employees and new versions of test tools
	Test tool upgrade	#licenses times updates	Costs linked to purchasing and installing test tool upgrades
	Integrations	#integrations costs per integration	Expansion and maintenance of integrations surrounding systems

IV. BUDGETING FUTURE PROOF ELEMENTS

This information is partly delivered by the overarching test procedure on company level. You can think of frameworks for reusability, tooling, test data generation for repeatability and a wiki for setting up the transferability aspect.

The question is which percentage of the required test capacity for test automation has to be reserved for developing future proof automated test scripts?

The following rules of thumb can be applied as shown in Table IV.

TABLE IV: RULES OF THUMB ON DETERMINING FUTURE PROOF FACTOR

Aspect	Priority	Factor
Reusability	H (= High)	1.2
	M (= Medium)	1
	L (= Low)	0.8
Repeatability	H (= High)	1.2
	M (= Medium)	1
	L (= Low)	0.8
Transferability	H (= High)	1.2
	M (= Medium)	1
	L (= Low)	0.8

An example:

100 hours have been calculated for test automation. In order to set up future proof test automation, the following values have been agreed upon (see below). Determining these values is done in accordance with the principal and is linked to a company's objectives.

Aspect	Factor
Reusability	H
Repeatability	M
Transferability	M

The number of hours required to set up this part future proof will be:

(100 x 1.2) x1 x 1 = 120 hours.

Another aspect to take into consideration is the scope of test automation. For which test type [3] are the test automation scripts developed? A sprint, an integration test (IT) or a chain test (CT)? The bigger the scope, the more synchronization with all parties becomes necessary. Think of which test data to use, which test scripts, availability of test environments for example and analysis of the results [17] [18].

You can introduce another factor namely the test type with the following parameters:

Test type	Factor
Sprint	1
IT	1.2
CT	1.5

Say you would like to do for example a chain test automation. The necessary effort, based on the table above would be:

(120 x 1.5) = 180 hours.

As stated, these are rules of thumb, which will have to be tested and adjusted by collecting data from yet to be executed case studies.

V. BUDGETING TEST AUTOMATION

In previous sections, it has been discussed both which general elements influence the test automation budget and that making test automation future proof also impacts the budget.
So how do you really budget test automation?
The following methods of budgeting will be elaborated on:
1. Percentage of the available time;
2. Pokering the required effort;
3. Pokering the required effort in combination with being 1 sprint behind.

A. Percentage of the available time

This method uses reserving a percentage of the total available test time for test automation as a starting point. A frequently used percentage, distilled from various projects, is

20% of the available test time. Suppose that for 100 hours of testing time 20 hours are used to do test automation. A part of these 20 hours is then used to make the test automation future proof.

By monitoring the actually needed capacity during each sprint, a realistic percentage can be established eventually. The velocity [19] becomes more and more accurate. The question is how reliable such a number is? Does a fixed number allow you to automatize everything that has to be automatized?

The risk is that in, for example, a sprint, not everything can be tested automatically, since the amount of work requires more time than can be realized in the time that is available. A debt is build up which either has to be removed during a next sprint, or in order to finish the amount of work scaling up is done. One of Agile's features is a shared team effort. Developers can support in test automation but this will be at the expense of other work, which puts pressure on the velocity and leads to not being able to realize all the selected product backlog items.

The way of budgeting, as described here, is a very basic way of budgeting, which begs the question of how much functionality can be tested automatically given the framework conditions. The advantage is that you always know how much time is available for test automation.

B. *Pokering the required effort*

Another way of budgeting is applying poker planning [20] specific for test automation. Perhaps initially this might seem like reserving a percentage of time. Initially, pokering the effort uses the brain power of the entire team to reach an actual estimation of the required time.

By placing the items which qualify for test automation on the product backlog, insight will be given into the amount of test work which has to be automatized. Pokering items also provides insight into whether the amount of work fits the current sprint. If the necessary effort is large one can decide to develop less functionality, so that developers can assist in developing the necessary test automation.

This way of budgeting has some caveats to take into consideration. Is the to be test automatized item on the backlog of sufficient depth to determine the scope properly? The second observation has to do with the stability of the features for which test automation has to applied. Is the team only capable of automatizing the test on unit level or also all the features itself?

C. *Pokering the required effort in combination with being 1 sprint behind*

To obtain a larger predictability of the work that has to be done, you can choose to start the test automation in the next sprint using the version of software which was produced in the previous sprint.

This approach has a number of advantages. The software which qualifies for test automation has reached a level of stability which makes it suitable for test automation. Another major benefit is that more detailed information is available with regard to functionality. After all, software has already been produced, which makes it easier to determine which effort is necessary to automatize the tests. Counting the number of functions goes back to the method of budgeting as applied within the TestFrame methodology [17].

This way of budgeting overlooks an important Agile principle namely the fact that working software has to be produced at all times. As a team you cannot guarantee that all software from for example a sprint works, simply because you can no longer test everything manually.

VI. A DEVELOPED EXAMPLE

To give an idea of the various elements influence on the needed capacity, a fictive example has been developed.

BASIC REQUIREMENTS:

Activity	Required capacity in hours	Calculating factor
Testing	1000	
Way of budgeting: 1 (fixed percentage)	200	
Future proof: Reusable: H Repeatable: H Transferrable: L		 1.2 1.2 0.8
Scope test automation: chain test		1.5
Relevant general elements Additional training: License costs		4 persons 1 day €1000, -- per day 4 licenses €1000, -- per year
Hourly fee		€75

TOTAL AMOUNT:

Activity	Calculation	Amount
Required capacity	(200 x 1.2 x 1.2 x 0.8) x 1.5 x €75	€25.920
Training costs	((4 x 8) x €75) + 1000	€3.400
License costs	4 x €1000	€4000
Total costs:		€33.320

VII. COLLECTION OF THE DATA

In the previous sections a few methods are described to estimate test automation in an Agile context. Till now there is no real evidence which method is the best. A first attempt was made as described in Section II. To verify the described estimation method a new survey will be set up to collect the data based on Table V.

TABLE V: COLLECTION OF THE DATA

Project	Type of Initial cost	Prio	Continuity costs	SDLC	Estimation method	Estimated hours	Actual hours

The major problem with the first survey was the timeframe. It was to short to collect data from different customers and projects. The new survey will take place during a period of two years to collect the required data as a base for a proper analysis. At least the data of 100 projects will be collected.

VIII. RETURN ON INVESTMENT

Initially, test automation costs money. As indicated, various elements have to be put in place before test automation can really be applied. The question is when the required investment will be recouped. Tied to this question is the question: what you will earn exactly? Soon thoughts will go to quantitative aspects. However, when it comes to return on investment (ROI) qualitative aspects also play a part. Table VI describes a number of aspects that show how you can recoup the investment.

TABLE VI; ASPECTS RELEVANT FOR THE ROI

Aspect	Description
Shortening test execution time [21]	Manual execution has been replaced by test tools by means of which test automation can be executed in so called off-peak hours. Besides this, a test tool is many times faster than a human being.
Prevention of regression	Because of the acceleration in test execution it has become easier to execute all automatized test scripts. Insight into possible regression can be gotten quickly.
Impact analysis in case of modifications	By executing automatized test scripts in the first sprint, insight into the suggested modifications can be gotten quickly. This can be especially beneficial in a Devops environment.
Time to market	By raising the test execution power, the company can enter the market much faster than its competitors.
Independence	By automatizing the functionality, a company becomes less dependent on a few functional experts. This expertise can be used in other parts or parts of which automation is not useful.
Reliability in the execution	The execution of the automatized test always happens in the exact same way. This provided insight into the stability of the software.
Uniform way of reporting	The test tool generates reports. These describe in detail what happened during the test execution. This makes it easier to track faults and takes less time.
Quality to market	The accuracy of tests gives a good insight into the quality and stability of the information system.

IX. CONCLUSIONS AND FUTURE WORK

This article describes three ways of thinking on how to budget future proof test automation in an Agile environment. These ways of thinking came into being because the existing methodology did not support a reliable budget sufficiently. They will have to be tried and tested in practice by means of a case study. Data has to be collected in order to eventually develop a balanced way of budgeting. Lastly, the article describes the benefits of applying test automation in an organization.

ACKNOWLEDGMENT

I would like to thank Marcel Mersie and Danny Greehorst for their contribution and giving me some of their precious time. Furthermore, I would like to thank the companies where I could set up test automation and develop the in this article mentioned three ways of thinking.

REFERENCES

[1] J. van Rooyen, “Effort estimation test automation in an Agile environment,” Valid2016, 2016.

[2] E. van Veenendaal, ”The Testing Practitioner, hoofdstuk 1,” UTN, 2002.

[3] E. van Veenendaal, M. Posthuma ’’Testwoordenboek, blz. 112,” UTN, mei 2010.

[4] M. Fewster, “Common mistakes in Test Automation,“ https://www.stickyminds.com, 2001, 2018.06.15.

[5] Agile manifesto: Agile manifesto, [online]. Available from: www.agilemanifesto.org, 2018.06.15.

[6] S. Hoogendoorn, “Dit is Agile,” Pearson, 2012.

[7] M. Fowler: Continuous Integration, [online]. Available from: https://martinfowler.com/articles/continuousIntegration, 2018-06-15.

[8] Blazemater: The advantages of Manual vs Automated Testing, [online]. Available from: https://www.blazemeter.com, 2018.06.15.

[9] B. vd Burgt, I. Pinkster, “Succesvol testmanagement, een integrale aanpak, blz 110-111,” tenHagenStam, 2003.

[10] D. Greefhorst, M. Mersie, J. van Rooyen, “Principes van testautomatisering,” Computable, 2015.

[11] DJ de Groot, “Testgoal,” SDU, 2008.

[12] J. vd Laar, “Technieken voor plannen en begroten van test projecten,” Testnet voorjaarsevent, 2009.

[13] M. Karlesky, M vd Voord, “Agile projectmanagement,” Embedded systems conference Boston, 2008.

[14] ISBSG, “ISBSG database,”

[15] Guru99: Automation Testing for Agile methodology, [online]. Available from: https://www.guru99.com, 2018.06.15.

[16] European Commission: GDPR, [online]. Available from: https://ec.europa.eu/info/law/law-topic/data-protection, 2018.06.15.

[17] C. Schotanus et al, “Testframe, hoofdstuk 6,7,8,” Academic Service, 2008.

[18] M. Siteur, “Automate your testing, sleep while you are working, blz 143-144,” Academic Service, 2005.

[19] Improvement: Term velocity, [online]. Available from: www.improvement-services.nl, 2018.06.15.

[20] Improvement: Term pokerplanning, [online]. Available from: www.improvement-services.nl, 2018.06.15.

[21] Test Automation: Test Automation Snake Oil, [online]. Available from: www.satisfice.com, 2018.06.15.

The CoWaBoo Protocol and Applications: Towards the Learnable Social Semantic Web

Athanasios Priftis
Information Systems Department
HESSO / HEG-GE,
Geneva, Switzerland
e-mail: athanasios.priftis@hesge.ch

Jean-Philippe Trabichet
Information Systems Department
HESSO / HEG-GE,
Geneva, Switzerland
e-mail: jean-philippe.trabichet@hesge.ch

***Abstract*— Web platforms and applications are generating a normalized environment for users to consume information. This process of making the Internet experience "clickable" and "fun" comes at a price: we are less inclined to face important decisions on how applications work, data are handled, or how algorithms decide. This article will examine the possibility of shifting from predetermined results to open and descriptive applications. We inscribe this effort in the context of the Social Semantic Web (s2w) and aim to add a pragmatic approach relying on the importance of humanly created semantics, as a means to fulfil the vision of the s2w. To achieve this we have introduced a descriptive protocol (CoWaBoo) that revisits fundamental web user activities such as search, classification, group formation and valorization of participation. This article will build on the 2017 results from our university group course and, particularly, the prototype applications created through the API of CoWaBoo during the same period. Our aim is to shift our attention from how things end up on the web, to how things *become*, regarding software and applications development. The conclusions of this article will provide further questions for the development of the protocol (CoWaBoo), its applications and the competencies that we need to develop to become actors in the s2w.**

Keywords-social semantic web; learning protocols; learning driven applications; semantic structure; descriptive applications.

I. Introduction

The growing web and algorithmic reality, where users perform prescribed functions or tasks, is an area that we need to continuously discover and understand. As we will attempt to demonstrate in this paper, the CoWaBoo protocol empowers a human-driven interaction in the web – social space [1] providing a better understanding of the opacity and the impact of these tasks, framed around the term 'digital inequality'. The latter has frequently focused on the distribution of computational resources and skills [2]. How people may be subject to computational classification, privacy invasions, or other surveillance methods in ways that are unequal across the general population, could be in violation of existing regulatory protections [3].

The widely spread culture of opacity of web applications, data or machine learning algorithms is operating a certain normativity: when a computer learns and consequently builds its own representation of a classification decision, it does so without regard for human comprehension. The examples of handwriting recognition and spam filtering illustrate how the workings of machine learning algorithms can escape the full understanding and interpretation by humans, even for those with specialized training [4]. Algorithms, such as those underlying the Google search engine, are often multi-component systems built by teams producing an opacity that programmers who are 'insiders' to the algorithm must contend with as well [5]. The opacity concern arises in the middle of an input - black box - output approach. For the most part, we know how the data are being fed into the algorithm: we produce it ourselves through our activities. We know some of the outputs of the algorithm and can reasonably infer how the algorithm has classified the data. What we do not know is how the 'black box' operates, or which bits of data the algorithm selects and how it uses that data to generate the classifications.

The elements above interact with our understanding of the Social Semantic Web (s2w). S2w can be defined as a space in, which social interactions lead to the creation of explicit and semantically rich knowledge representations. An interaction of collective knowledge systems is able to provide useful information based on human contributions and gets better as more people participate [6]. As web application development is conditioned from the communication structure and technological decisions of the organizations [7] providing them, alternative approaches become crucial.

To sum up, we cannot rely on the modern disciplinary methods and frameworks of knowledge in order to think and interpret the transformative effect which new technology is having on our culture. It is precisely these methods and frameworks that new technology requires us to rethink [8]. Our goal in this paper is to propose an analysis that intersects the current state of opacity and, at the same time contributes to our understanding and application of semantic rules to user created applications.

The following sections of this paper will pursue this analysis. In Section II, we briefly position protocols as an important, but not sufficient, parameter against an opaque, black-box culture of application use and development and introduce the Social Semantic Web mechanisms. Section III is devoted to CoWaBoo, a protocol of building web applications, empowered by specific rules and architecture.

In Section IV, we demonstrate the prototype applications, built by students as proof of work for creating open and descriptive applications, based on the CoWaBoo API. Finally, in Section V, we highlight how such user driven applications validate the rules and functions of the CoWaBoo protocol and how more complex social rules can emerge, while raising further research questions. Our results include presenting the use of CoWaBoo for a specific, knowledge management, application, providing with some early directions on social tagging and user based web ontologies.

II. PROTOCOLS, DATA AND THE SOCIAL SEMANTIC WEB IN A WORLD OF INTERNET APPLICATIONS

The advent of blockchain facilitated the creation of several protocols allowing for the implementation of user driven web applications under the heading of decentralized applications [9]. The most prominent example of such protocols is Ethereum an open-source, public, blockchain-based distributed computing platform, featuring smart contract (applications) functionality [10]. A protocol is an ambivalent space where both opacity and transparency are possible and certain short-term goals are necessary in order to realize one's longer-term goals [11]. Applications running on protocols can use or, sometimes, play down this contradiction but never break away from it. A protocol is never neutral in the sense that all decisions regarding its functions are already set to bring some kind of normality on users (agents) behaviors. In this sense, it contributes to generating a semantic normativity for users to consume information, but not necessarily transforming it. This process is interlinked with the production and reuse of data in semantic mechanisms that are prescribed in accordance with the objectives of their creators.

This use of semantic mechanisms to describe the meaning of data, and enable the description of their metadata, is allowing for more complex user driven analysis. This includes using potentially any data in any format and apply smart data management algorithms as a commonly accessible resource [12]. However, this initial position does not explain how the social happens in the semantic. Having semantically structured data inputs does not guarantee that a social learning is in place or, even, possible. We will turn to the Social Semantic Web (s2w) to understand how this could possibly happen. S2w aims to add a pragmatic approach relying on description languages for semantic browsing using heuristic classification and ontologies, emphasizing the importance of humanly created loose semantics as means to fulfil the vision of the semantic web. Manuel Zacklad, Jean-Pierre Cahier and Peter Morville define s2w as the pace layering of ontologies, taxonomies, and folksonomies to learn and adapt as well as teach and remember [13].

Various investigations have been published regarding the evolution of the social and semantic web, including the development of ontologies for tagging [14], the extraction of ontologies from social network graphs and folksonomies [15], collaborative ontology evolution and reasoning over tags [16]. The social and semantic web is appearing as a mixture of multi-disciplinary information that is evolving within an open environment [17], making it difficult to initiate a learning process. In these efforts, there are various possible social approaches for solving the problems of user driven ontology evolution for the semantic web. Users could create folksonomies or flat taxonomies to document their information. Social Network Analysis (SNA) allows the ontology to be extracted from the tags and be reused into topic maps or ontology stores. These organized tags could be manually analyzed to create a more sound ontology. Another approach is to create a system for self-governance where the users themselves create the ontology over time in an organic fashion. All of these approaches could start out with an empty ontology or be seeded manually or with an existing ontology.

The protocol that exemplifies user-oriented ontologies is SSWAP (Simple Semantic Web Architecture and Protocol). SSWAP is an architecture and protocol for semantic web services. It uses the W3C standard of OWL (Web Ontology Language) to ground a web services model on computable semantics and logic by formulating an architecture and a protocol. The HTTP API allows developers to generate SSWAP RDF/XML graphs using JSON [18]. The main application, based on SSWAP, is called iPlant. The iPlant Collaborative seeks to enable data-driven scientific integration both within the enterprise and across web resources, including widely used programs of general interest and niche programs for specific needs [19]. IPlant's Semantic Web Platform is developed as the technological conduit for integration across various plant resources. It is by having software layers handle data, service syntax and semantics that iPlant seeks to free the scientist to focus on data and service use.

Application development comes at the junction of these diverse and interdisciplinary efforts. With the multiplication of user driven applications, we could tap into a large part of the motives and rules of the created interactions. When it comes to information and internet technologies' mediated systems, the technical arrangement of the underlying interaction platforms do have an impact on the social cognition processes [20]. The process of channeling more or less sophisticated representations, from opinions and likes/dislikes, to tags, sentences, or documents are some of the elements that need to be, continuously, examined.

However, a single instance of data (i.e., a tag) can never be semantic on its own. Semanticity is captured at the logic of organizing data within a system of relations or rules applied to this system. Several problems encountered in ranking, documentation and automated search can be handled by semantic conceptualization and graph theory [21], while many others persist. Our position is that instead of relying entirely on automated semantics with formal ontology processing and inferencing, humans could (learn to) build their applications. While the semantic web enables integration of business processing with precise automatic logic inference computing across domains, a Social Semantic Web could be used for a more socially oriented technology interface, allowing for a more divers interaction between various objects, actions and their users. The CoWaBoo protocol, including this paper, aim to contribute to this exact area, in between semantically oriented protocols such as SSWAP, or transaction oriented ones like Ethereum.

As importantly, we need to connect s2w with a clear educational demand; the people who craft the code, that determines all the million material ways in which we interact, are able to consciously articulate the things they believe, agree or disagree [22]. The Social Semantic Web presents us with a possibility to understand and affect the design of software and applications and, eventually, how could it be reformed. The CoWaBoo protocol and its applications are set to intermediate this effort.

III. THE COWABOO PROTOCOL AND ITS OPEN API

CoWaBoo starts as a concept, leading to an ambition of understanding and affecting user actions in a socio technological context. The CoWaBoo protocol considers the addressee, user of these wider computational systems made up of processes, technologies or networks, as a possible actor in their design. As the interaction needs to be open to further development, the code of the CoWaBoo protocol needs to apply the following default rules [1]:

- Assure the lucidity of past (as stored data), present (as current data collection, or processed archival data), and future (as both the ethical addressee of the system and potential provider of data and usage).
- Store objects with a semantic description that generate functions for applications.
- Demonstrate that the code's mediation can be rethought, researched for intervention, contestation, as a tool for the un-building of other code/software systems.

The overarching concept of the protocol is to formalize an always-editable space realized by the rules set above. However, this space stays vague if we do not test and experiment its utility. This is why we bring in the collective observatories application, as one of the many possible applications, built upon the protocol. In this application, the protocol would store and allow us to recover a general index of a given subspace (observatory) and then be able to navigate through the different versions of the entries in this observatory. To achieve this within the application, we introduce two more layers of representing information reusing data from the protocol: a) the use cases, or the way we propose users to explore the possibilities of the protocol, through the creation of collective observatories and b) the graphic representation of classified information. The use cases need to be concrete with a measurable result. Users can search information that communities have already curated and form groups. The graphic interface attempts to address how users will experience the above.

In terms of communication, data and account handling the CoWaBoo protocol adopts the following approach. All data are stored on the InterPlanetary File System (IPFS), a P2P (peer to peer) storage protocol, with its current state available in the application. Account creation is based on Stellar, a blockchain based, open source protocol, used for all transactions between users. CoWaBoo is utilizing a NodeJs server with a Stellar Javascript SDK to provide the CoWaBoo API with a way to communicate directly with Stellar. Every time that a new member is subscribed, a new Stellar account is created. This account receives the minimum amount of lumens (Stellar currency) to work properly as a CoWaBoo account. Once the account is created, it automatically give its consent to carry out currency exchanges created by the "bank", or main wallet, of CoWaBoo in Stellar. Consequently, all transactions (i.e., votes in groups) are stored on the Stellar blockchain infrastructure creating an exchange community with a cryptographic Public Address and Secret Key. While Web 2.0 applications tend to prescribe our participation, in CoWaBoo we seek to re-open the discussion on the group rules and their results. Therefore, all new entries or definitions are openly editable, as long as her/his entry is voted in a group. In the following paragraphs, we will discuss in detail the utility of this function and demonstrate if and how the default rules are applied.

Once an entry is added, modified or deleted, the observatory generates a new version of itself with a unique, cryptographically generated and traceable identity. This system of versioning, creating new instances for every edition on an entry, follows the blockchain paradigm. Blockchain as a distributed, cryptography boosted, database technology is a thing of the 80s, which computational capacity of our time brought to full implementation with the Bitcoin deployment. Blockchain can be understood as an implementation of distributed ledgers that comes with a unique set of possibilities in its design. It opens up the way to shared databases, where multiple entities can transact, with no or some trust between them, co-existing with no intermediation.

The CoWaBoo protocol reproduces the main blockchain synchronous properties, as described above in the following generic approach: a) accessible and affordable shared data with resilience through replication and no single point of failure and control, b) where multiple entries are possible, c) based on the possibility of disintermediation. Blockchain implementation comes with more interesting feature: d) application based transactions, or smart contracts [23]. Going back to our collective observatories application, we will try to point out the exact process and code that demonstrate the above. The CoWaBoo objects described by the protocol are organized in the following categories:

- Dictionaries or Observatories
- Entries
- Propositions
- Users
- List of all users
- List of all tags

The process of communication and connection between these objects in the protocol is depicted in Figure 1.

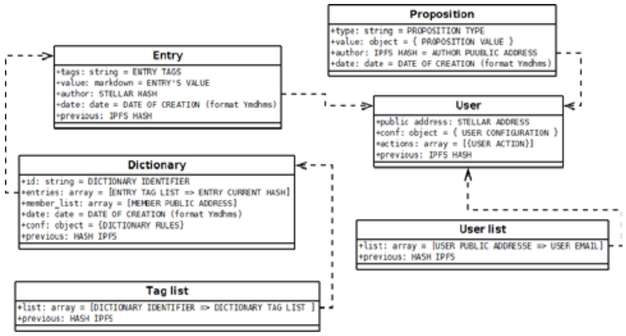


Figure 1. The connections between the CoWaBoo protocol objects

This loose semantic architecture of the protocol, as well as, the collective observatories application proposed above have one important result. CoWaBoo can be understood as a re-documentation effort opening to possible, iterative transformations of user drive applications through its API and the way it organizes the user access to its objects. Figure 2 offers an overview of the CoWaboo API and its structure: tags, observatories, entries, users and accounts creation. It consists of several functions and elements, also presented in the following chapter: The user-oriented view of the CoWaBoo API is demonstrated Figure 2.

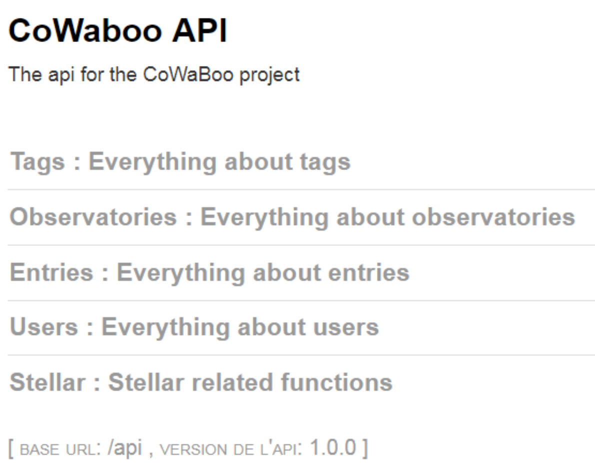


Figure 2. A user-oriented view of the CoWaBoo protocol and its API

In the following section, we will demonstrate how user driven applications are intermediated by the CoWaBoo API. This activity provides us with a key resource for understanding semanticity, as the logic of organizing data within a system of social relations. These relations will be deployed with user proposed rules functioning on code. Beforehand, we will present the use cases that define the protocol and present the context for the CoWaBoo API utilization. In other words, how it prescribes the construction of user driven applications with socio semantic decisions and contributions.

IV. Implementing the CoWaBoo Protocol through Use Cases and User Applications

A protocol is never neutral in the sense that all decisions regarding its functions are designed to bring some kind of standardization on its user's actions. Thus, it becomes crucial to describe, in more detail, the use cases of the collective observatories application and illustrate how the interaction with the protocol will take place. We can think of the use cases as a descriptive middleware, positioned between the protocol and the collective observatories application, appearing itself to its users, as its initial implementation.

Build your story: A prompt to click on the start button and move to an empty text area is the first step. The empty text area is destined to be the user's notebook, potentially filled up with search results discovered using the connected APIs for search on tag, bookmarks, articles and existing entries, themselves, linked to the visualize tags and stories in our observatories use case. Keywords for initial search lead to tags looking to visualize existing stories in the CoWaBoo observatories. The user selected data are inserted in the text area, as part of the actual user story. This tentative story-result can stay local, private and unfinished, which is posted in an observatory, or become the first entry in a new possible observatory. Adding a post in an existing observatory is subject to a verification process depending on the rules of the observatory: self – validated means that the post is validated by the user, peer – validated means validated by a "vote" from someone in its existing members.

Edit a story: A click on the full text search button of the application connects the user's keywords, then transformed to user tags, used to visualize existing stories in observatories or observatories themselves. These tags are linked to the visualize tags and stories in observatories use case. The CoWaBoo user can select them as content for further editing into the text area.

Start a community of transactions (group): A click on the community button initiates a community creation function with the possibility to add emails, each participation verified through the related email account and attribution of both public and secret keys. The group creation launches the possibility of starting: transparent, intra-group transactions with all group users being informed on energy limits to credit or stored value of the group. Group participants use the secret key to perform transactions, while the results of the transactions, as well as, the user balance remain publicly linked to each user's public key. New members propose themselves through direct demands to join an observatory (group) or through accepted stories: when observatory entries are accepted they become a member, while, his/her public key is added to the group users for further transactions.

Editing tags: CoWaBoo treats tags as distributed objects, recasting the tag object as an autonomous transaction providing its user with an opportunity to redefine, rebuild and redistribute through the work of others. Tags in CoWaBoo acquire a multiple meaning as they simultaneously represent:

a) “Tags to be”: user typed search keywords in the text area leading to tags used by other users.

b) Keywords that are then selected as tags, leading to educated choices of entries description, or proposed entries or definitions for observatories.

c) Tags are treated as semantic elements pointing to entries in observatories (list of tags per observatory).

d) Names of observatories are treated as tags (list of observatories). They are proposed to users for consultation before creating a new observatory.

Propose or “vote” an entry or a member (validation & valorization). Each story is accepted as an entry (definition) when posted and voted on by at least one group member as a verification (the fastest reply - user is considered for attributing the value of the transaction). Each accepted entry is tokenized with one (1) energy (limit +10 for every user), while a vote for an entry uses 1 energy, (- 10 for every user). The variation in personal energy is, initially, anonymous but transparent, connected to each user’s public key.

Visualize tags and stories in observatories: This use case is connected to user keywords (full text search) becoming tags and visualizing an index view with linked tags, entries and observatory names.

The following are some important questions that will guide the presentation of the initial results of the protocol and its collective observatories application: Firstly, do tags serve at the same time as descriptive keywords, linked data (to stories) and ongoing collections (observatories names)? How do they connect entries and observatories? Do they provide some kind of navigation through the information initiated by the application and stored by the protocol? Secondly, the protocol does not promise, or highlight, a completed story or observatory but a possibility of creating stories and editing all products in future events. How is this appearing in the existing digital space? Can the CoWaBoo protocol and its linked applications serve as an experimental understanding at semantic, representational and process levels in a context of collaboration and knowledge production?

Before attempting to answer these questions, we will present how the CoWaBoo API prescribes the construction of user driven applications. As mentioned before, the CoWaBoo API allows users to experiment and develop applications on the CoWaBoo protocol. This process seeks to create a space where a user centered semantic construction becomes possible, under the defining and implementation of **tags, observatories, entries, users, blockchain enabled accounts and transactions**.

Tags: The user can call (GET) all the tags and access the documentation in the provided URLs. Tags, as semantic objects, have multiple functions: form an id for entries, act as descriptive keywords or provide a list of data available. Figure 3 demonstrates how the use of a simple API function (GET) allows the user to call all tags with their semantic description, see exactly how this function works (CURL) for the CoWaBoo protocol and access the generated URL available to reuse.

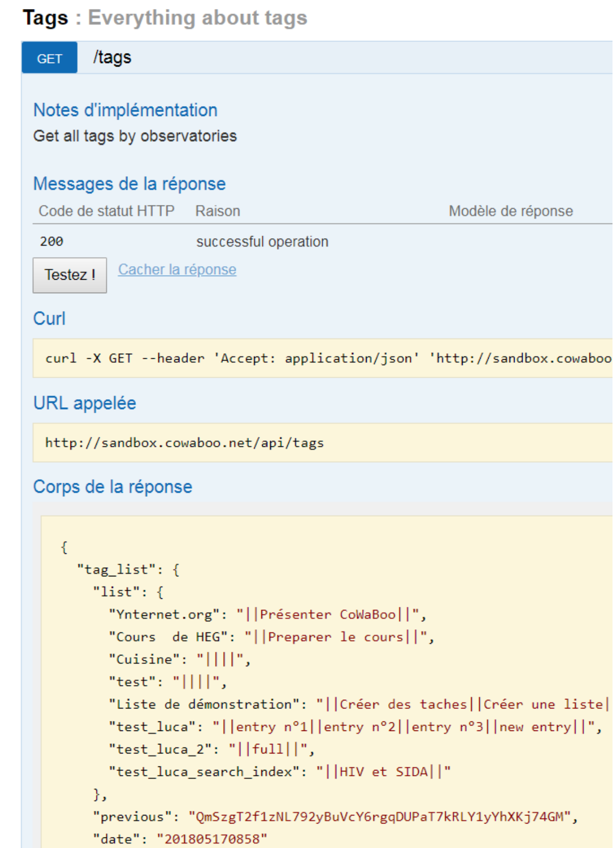


Figure 3. The CoWaBoo API call for all tags

Observatories: This API section acts as the main logical container of whatever information or concept is to be expanded by the forthcoming entries of the application. In a sense, this is an initial hierarchization and organisation of the application’s data as shown in Figure 4.

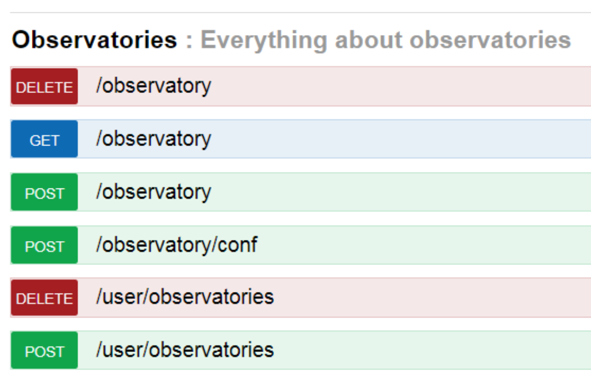


Figure 4. The CoWaBoo API observatories

Entries: The Entries section is the core semantic repository of the protocol. This is where JSON, XML or any other structure form of user-oriented and application rules data can be saved, recuperated, edited, deleted, reposted and, infinitely, versioned through its cryptographically created editions. Figure 5 summarizes the CoWaBoo API entries.

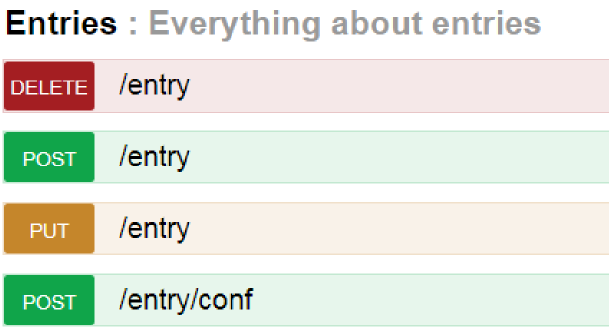


Figure 5. The CoWaBoo API entries

The Entries section (Figure 6) come with a detailed API description: this is where the user Secret Key (author) is used to add and validate data posts and add tags as entries i/d or keywords.

Figure 6. The CoWaBoo API entries' section structure

Users: Users, as shown in Figure 8, are handled by creating a new, public and blockchain (Stellar) linked, account. This creates a public and a secret key communicated automatically to every new user. Both are then reused to validate observatories, entries and perform transactions in applications (i.e., votes).

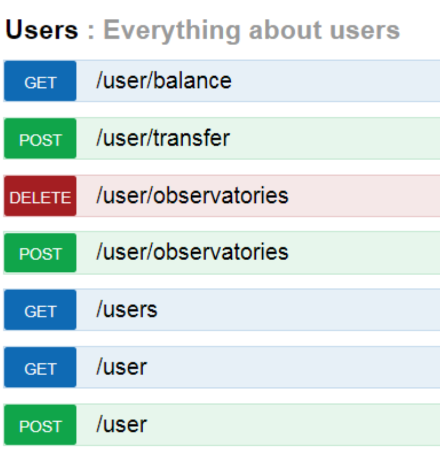


Figure 7. The CoWaBoo API users' section

Figure 8 demonstrates how the user's transactions are taking pace using their public and secret Key. The amount can be any value the user wants to transfer through the Stellar blockchain protocol. It corresponds to a generic currency called "energy" created by the CoWaBoo protocol and allowing users to define it as credit, votes or other. In summary, "Energy" is a subdivision of Stellar's Lumens cryptocurrency and is implemented as a rule centrally administrated by the protocol maintainers.

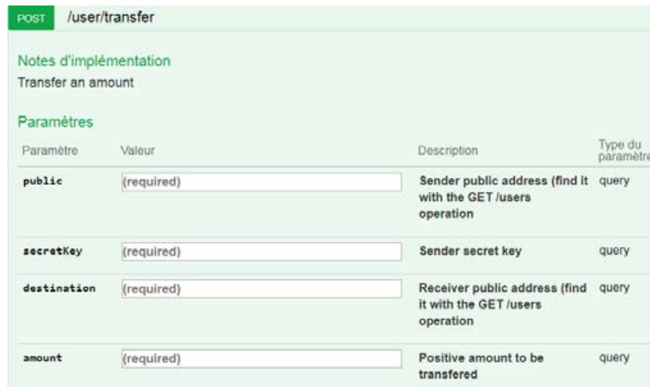


Figure 8. The CoWaBoo API and its users' transactions section

Stellar use: As mentioned above, Stellar related functions are adding the possibility to access user related transactions. This function, described in Figure 9, gives the possibility to users to create their own currency for their applications use and test all their transactions with the public Stellar blockchain.

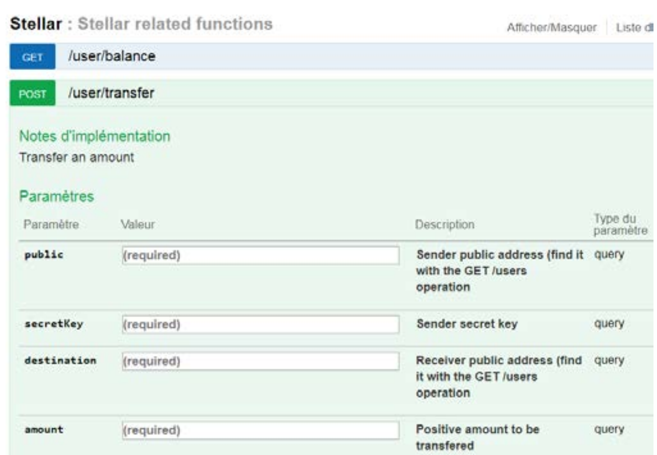


Figure 9. The CoWaBoo API and its Stellar related functions

Different instances of the API are easily created to provide a new space for users to develop their applications. At this point, every new CoWaBoo API instance is controlled centrally. However, this is not a bottleneck for future creation of applications because there are no data or other information kept locally and the creation of API instances could be handled directly by its users. This architecture, although rudimentary in its current state, provides a novel experience to users on creating new applications. Users are free from the data based infrastructure and account generation process to focus on the semantic representation of their application's data and processes.

V. Early results and remarks on how to evolve the social-semantic mechanisms in app building

Our results are organised and presented in the following areas: firstly, we present with some documentation around the learning and development context: this is a testing effort structured around groups, for a minimum prototype application building activity. Secondly, we will highlight an example of an application built in the CoWaBoo protocol context, demonstrating the social semantic potential of this process. Finally, we will try to summarize and organise our contributions to the development of a user driven s2w understanding.

The design and implementation of the CoWaBoo protocol started in 2015, as a research effort on building a bottom up, social bookmarking process. The design activities and results of our 2015 - 2016 experiments including focus groups with experts and students testing various versions of the application, led to the creation of the presented collaborative observatories application on the CoWaBoo protocol. This application was introduced in our university courses during the spring semester of 2016 for further testing and analysis. The use of the CoWaBoo protocol and API, as a user-driven, application development space was introduced in 2017. This new experimentation phase was organized with twenty-four students of the University of Applied Sciences in Geneva during a dedicated course on Digital Business Technologies. This group of, 3rd year, bachelor students were invited to work in groups of three, around a specific thematic area (i.e., Blockchain, Bitcoin, Wikipedia, Airbnb, Uber, Free and Open source software, Open licensing, Open innovation) using the CoWaBoo collective observatories application as the main space of documentation and the CoWaBoo protocol and API for their own prototype applications' development. This activity included two concrete areas for further evaluation. Firstly, participants were asked to organize and present an introductory course in class. The course is documented in the collaborative application based on CoWaBoo. Thus, it was important to demonstrate how the use of an application relates with the CoWaBoo protocol. Secondly, each group had to design and prototype an application based on CoWaBoo. While reviewing below these results, we will try to highlight several social semantic elements related to web applications' building.

In the following paragraphs, we are presenting selected results regarding the use of tags from students in their respective observatories. This is linked to the use of an application related with the CoWaBoo protocol and is connected to our Editing tags use case, described in Section IV. Our understanding of tags as descriptive keywords, linked data and ongoing collections is the basis of the following remarks.

The first remark targets the way the information of each observatory, eight in total, is presented, accordingly to a certain tags selection. Let us use the "blockchain" observatory to see how its tags are visualized:

Blockchain (theme and name of the CoWaBoo observatory) with selected tags:

|| DAO || France || IDE || Parlement || analysis || badge || banques || bdd || bitcoin || blockchain || chain || concept || crypto-money || ethereum || finance || finance on blockchain || fonctionnement || governance || infographie || peer-to-peer || plate-forme || politique || presentation || questionnaire || reference || smart contract || technologie || wikinomie || wikipedia ||

This static and hierarchized representation of an observatory with its related tags is a straightforward use of all social bookmarking applications on the web and gives access to related user entries.

The second remark is on how each tag is linked to other tags. This function uses data, coming from all other CoWaBoo observatories, connecting them as linked tags with the word blockchain. In this sense, we are advancing to a networked relation and navigation of various observatories through commonly used tags.

Linked Tags (in English and French) to other observatories for the "tag" blockchain include:

Cowaboo, properties, avantage, bitcoin**,** monnaie**,** concept**,** fonctionnement**,** infographie**,** smart contract**,** reference**,** presentation, wikinomie**,** badge**,** questionnaire**,** Wikipedia**,** peer-to-peer**,** innovation ouverte**,** plateformes**,** copyleft**,** creativecommons**,** smartcontract

A third remark has to done on the entries (stories) that are being created and edited in the respective observatories, with the tag blockchain in each entry. Entries using the tag blockchain include:

- CoWaBoo

|| CoWaBoo || blockchain || properties ||

- Bitcoin

|| avantage || bitcoin || blockchain || monnaie ||

- Blockchain

	blockchain		concept		fonctionnement		infographie	
	blockchain		smart contract	
	blockchain		reference	
	blockchain		presentation		wikinomie	
	badge		blockchain		questionnaire	
	blockchain		smart contract		wikipedia	
	bdd		blockchain		peer-to-peer	

- Innovation Ouverte

|| R&D || blockchain || innovation ouverte || plateformes ||

- Copyleft

|| blockchain ||copyleft || creativecommons || smartcontract ||

In this representation, we observe that multiple tags act as the title of an entry. The word blockchain is now used as a search keyword that leads, subsequently, to entries that include it in its tags.

A fourth remark needs to be placed around the use of the word blockchain in all the observatories that CoWaBoo is hosting. **Observatories using the tag CoWaBoo include:**

- CoWaBoo
- Bitcoin
- Blockchain
- Innovation Ouverte
- Copyleft

This presentation highlights the observatories that can interest a user with a search interest in the blockchain: before clicking on a specific entry, the CoWaBoo user has a hierarchized overview of the areas that could be connected to his/her research.

These four remarks derive from the daily, dynamic use of the CoWaBoo protocol and its social bookmarking – collective observatories application. They summarize the possibility of an application, like our collective observatories, to use the CoWaBoo protocol's functions in order to reclassify user data. The protocol therefore formalizes all the necessary functions:

- Data storage, recording new data in an observatory: entry, member list, configuration
- Ensuring the current state of an observatory
- Possibility to add, change and delete data from an observatory
- Registration and acceptance of new members
- Vote counted and executed through blockchain intra-community transactions
- Configurability and re-applicability of the rules of the group in the protocol itself.

It is important, to describe if and how the default rules are applied. Once an entry is added, modified or deleted, the observatory creates and stores a new version of itself. In Figure 10, we depict the current state of the blockchain observatory. This includes its i/d, entries, members, date, configuration (public or private) author and, most importantly, the path (hash) to its previously stored version. This process applies to all data stored from the CoWaBoo protocol and generated from the collaborative observatories application: observatories, entries and member list.

Figure 10. A CoWaBoo (blockchain) observatory stored on IPFS

As already mentioned the InterPlanetary File System (IPFS) is a content-addressable, peer-to-peer hypermedia distribution protocol. Nodes in the IPFS network form a distributed file system. IPFS plays a crucial role in CoWaBoo acting as a public ledger of all posted or edited data. The same goes for any rule, or post. All changes in membership, authorship and rules of this observatory are re-traceable in the blockchain logic of the protocol with the previous version always available. Content wise, things are significantly different, compared to a standard social bookmarking application. Entries are taking more of wiki form with a descriptive text and links. Figure 11 shows the entry linked to the following tags:

|| France || Parlement || banques ||finance || politique || technologie || entry

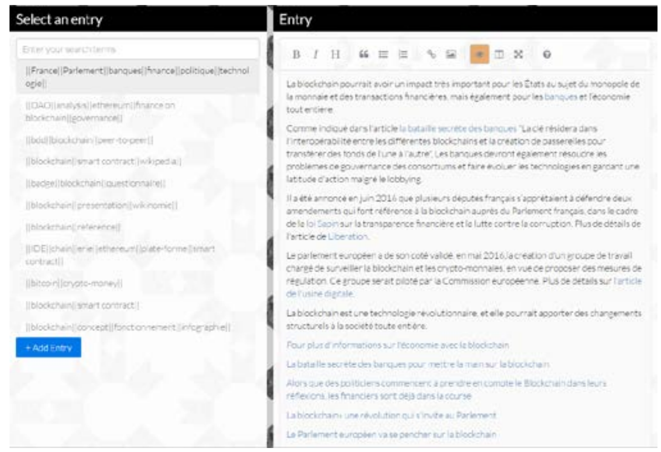


Figure 11. A CoWaBoo, collective observatories application, entry

There is a clear shift, in all observatories, to adding text and sources in entries and forming them more as stories. This is due to the discussion related to CoWaBoo lecture on social bookmarking and the evaluation criteria but it goes hand in hand with our initial anti-click-culture approach. In the table below, we summarize how CoWaBoo introduces the redocumentation process in steps.

TABLE I. THE COWABOO REDOCUMENTATION PROCESS

Step title	Description	Process
Search and Post	Users are invited to search in existing communities, discern knowledgeable others, edit text and post result in existing groups	Search through tags on existing community curated resources Select resources and edit a story Publish on a group a shared "block of information" with tags ⇒ Create unique link (URI) that could be posted on selected web services
Reuse and Curate	Initiate a process of critical scrutiny and group content production for participants	Engage in a group - observatory, where an exchange between users, data and group rules is possible. Provide feedback on group's entries Contribute to the implementation of the group's collective observatory Provide history of group decisions and documentation

The second area of results announced in the beginning of this section is around the social semantic web elements in application building possibilities of the CoWaBoo protocol.

The user applications created in 2017 complete the results presented above. The overall estimated effort to create each application is one week, as a total effort for all group members. The main idea behind this initiative was to introduce them to an early application building and confront them with the design and implementation opportunities and decisions that come therewith.

To illustrate our analysis we will use the ShareMyPlace app as a representative example of the eleven applications created and available in the CoWaBoo applications repository. The landing page of this application is presented in Figure 12 and acts as a list of private parking spaces available in the city. The use of this application can see the entries involved (time, locality, contact) and reserve its place.

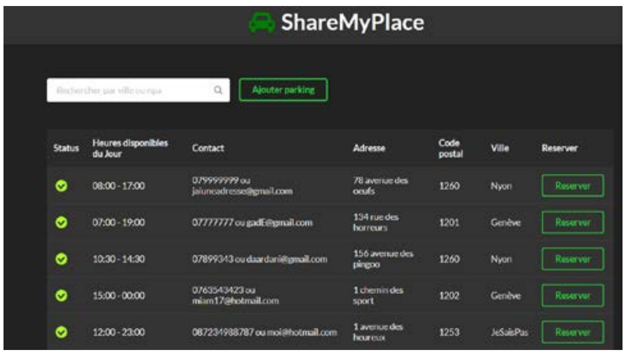


Figure 12. The ShareMyPlace CoWaBoo application: landing page (list observatories)

The first observation that we can do while clicking on the Reserve button is that we can "pay" in credits. These credits are blockchain transactions linked to the CoWaBoo protocol possibility to use its currency as credits or votes. Once the function triggered, the user needs to identify as a blockchain user and complete the transaction. Figure 13, shows the interface of the ShareMyPlace CoWaBoo application.

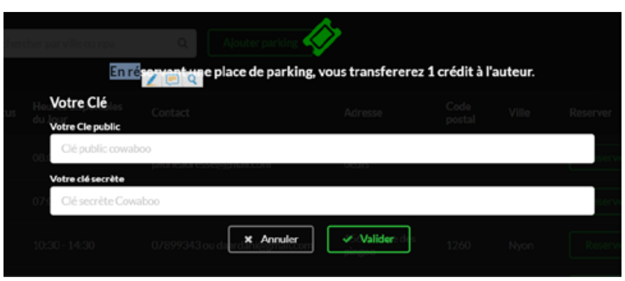


Figure 13. The ShareMyPlace CoWaBoo application: transactions' interface

The second observation is about the use of the protocol to organize the post of entries in the application. As shown in Figure 14, entries can have multiple values and connect to observatories. Their posting to the application involves a Secret Key validation, meaning that users need to identify, describe and propose a new parking space as a tentative or permanent entry (transaction) to an observatory.

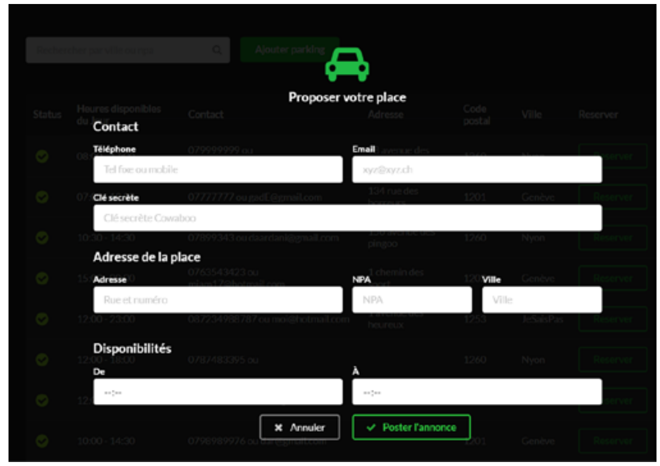


Figure 14. The ShareMyPlace CoWaBoo application: entries

The main social semantic lesson to retain from this prototype application development is the way its observatories (dictionaries for the protocol) are organized. The application designers opted for the creation of multiple observatories: one observatory was used for each parking space proposed in the application. Thus, the developers preferred to make a distinctive and separate entry for each value leading to a linear, heavy and non-efficient creation and data storage process, as shown in Figure 15.

dictionary:{id: "parking1ede38f6-446c-002e-1cc9-2493858aba0e", entries: {,…},…}
dictionary:{id: "parkingcdd6d6f1-5987-911b-ea45-9cdff84a68b1", entries: {,…},…}
dictionary:{id: "parking0906c3b5-6f17-0c10-58d9-11ebf252e77d", entries: {,…},…}

Figure 15. The ShareMyPlace CoWaBoo application: multiple observatories, one observatory used for each parking space

Multiple entries can be inserted in multiple observatories, as shown in Figure 16. This implies a minimum of effort to organize and describe the application data and rules in a coherent and structured way. This would mean, for example that one observatory could accommodate all values of an entry through a semantically structured text.

dictionary:{id: "parking0906c3b5-6f17-0c10-58d9-11ebf252e77d", entries: {,…},…}
author:xxxx-xxxxxxxxx@etu.hesge.ch
conf:[]
date:"201705231318"
entries:{,…}
id:"parking0906c3b5-6f17-0c10-58d9-11ebf252e77d"
member_list:["xxx.xxx-dasilva@xxx.hesge.ch"]
previous:"QmPhqxXC6xkLWYwVvpnjUboyEtGaSxngyStdjUpsPff9Gg"

Figure 16. The ShareMyPlace CoWaBoo application: multiple entries in multiple observatories

We will profit from this lesson learned in order to provide an analysis of a different experience of application building through the CoWaBoo protocol. This example falls under the social semantic web elements in application building possibilities of the CoWaBoo protocol but comes from a personal student effort during its Bachelor degree work during the summer of 2017. This new application is called GraphtoLearn and it is being designed as a learning path based on the indexing and organisation for several keywords. It starts out as a thesaurus of learning courses in the Information Technology area, augmented with the following possibilities:

- custom and dynamic term search based on user participation (user and entries reputation and institution rules)
- user participation and entries annotation comes with assigned reputation rules
- 3D visualization of the results search that attempts to demonstrate the above rules

This indexing - learning application demonstrates the rise of social semantic structures empowered by the CoWaBoo protocol. We do not intend to analyse all the GraphToLearn possibilities but focus on the organisation of entries and reputation, rules as a part of its social semantic web elements.

Firstly, let us have a look on Figure 17 and, particularly, on how the architecture of GraphToLearn is organised. The API CoWaBoo is central in all data storage and reputation rules (votes on user actions and entries). GraphToLearn comes with the ambition that its semantic entries can describe and incorporate existing structured data; new user generated data and automatically harvested data (i.e., Google, Wikipedia). The selected data flows can be used to enrich the GraphToLearn entries, as well as, boost the application's reputation and search algorithms.

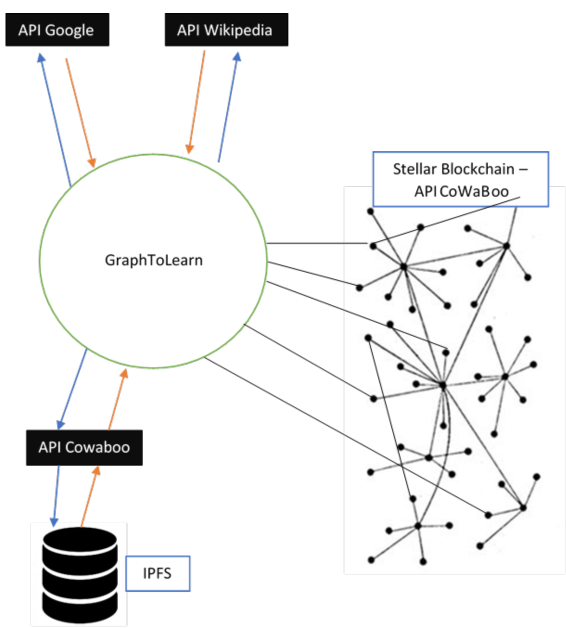


Figure 17. The GraphToLearn initial architecture

In this architecture, GraphToLearn is presented as a concept where the aforementioned possibilities are inscribed. This is a crucial step to our analysis, a clear intention to explore the social semantic in knowledge organisation and production. It shows that the application developer is inspired by the CoWaBoo protocol to deploy a more complex vision of user-oriented taxonomies, including a system for their self-governance. The organization of its entries, as presented in Figure 18, take a semantic turn with the following choices:

- One observatory including all entries
- Each entry includes all the needed information in a JSON format

```
{
  "dictionary": {
    "id": "Words",
    "entries": {
"QmST3o7XqmJLG8ZHa1M81vFH1ZHgAocammPEjBsR9szaho": {
      "tags": "||ActiveX||", -> nom du terme
      "value":
"{\"name\":\"ActiveX\",\"type\":\"Terme\",\"source\":[\"CRI\"],\"definition\":\"\",\"explications\":\"Contrôles utilisés en programmation web pour permettre d'animer des pages.\",\"context\":\"Technique\"}",
      "author": "xxx@etu.hesge.ch", -> la personne qui l'a inséré
      "date": "201707101559", -> date de création du terme
      "previous": null,
      "conf": []
    }
  }
```

Figure 18. The GraphToLearn: observatory and entries

Each entry is structured and described as presented in Figure 19. Various elements (data with their metadata) of the application are described and posted in the "value" area, following the design of the CoWaBoo API.

```
"QmShYR6hQrD83pqh7ndofhi8ipQ7QC7w6t6N95dMn13Soe": {
  "tags": "||MIB||",
      "value":
      "{\"name\":\"MIB\",\"type\":\"Acronyme\",\"source\":\"IDEC Metafichier\",\"modules\":\"471,498\",\"definition\":\"Management Information Base.\",\"explications\":\"Arborescence de variables stockant les états des matériels et logiciels selon le protocole SNMP.\",\"context\":\"Technique\"}",
  "author": "xxx@etu.hesge.ch",
  "date": "201707111415",
  "previous": null,
  "conf": [] -> qu'est-ce que je pourrais mettre dans conf ?
}
```

Figure 19. The GraphToLearn entry structure

Figure 20 demonstrates, through a JSON representation and structure, how the value of the entry is organised and posted in a specific entry (see Figure 20).


```
"{
  "name":"MIB",
  "type":"Acronyme",
  "source":"IDEC Metafichier",
  "modules":"471, 498",
  "definition":"Management Information Base.",
  "explications":"Arborescence de variables stockant les états
   des matériels et logiciels selon le protocole SNMP.",
  "context":"Technique"
  "commentary":"",
      "review":"",
        }"
```


Figure 20. The GraphToLearn JSON representation (data in a single entry)

There is a second area, where the socio-semantic intention of the developer appears in GraphToLearn: the reputation rules for both users and entries. We will present, briefly, the reputation rules, not as a complete technical approach, but as a token of the reflection that the CoWaBoo protocol has allowed for.

Setting up the GraphToLearn reputation makes an algorithmic search possible, allowing the application to sort out the proposed terms and highlight terms that have a better reputation in its search result. This element improves search by distinguishing terms that are related to the term the user wanted to search for. It is reducing search time by looking at the reputation of each term that would be best suited to the term initially sought.

To implement this system of reputation GraphtoLearn defines certain criteria in order to be able to evaluate each term (Term based reputation). It sets a corresponding weight for each criteria. The weight allocation is based on the sub-division of the CoWaBoo protocol currency. The main principles of the user reputation include modifications, new contributions, validating, or not, the work of others. A user who proposes an accepted entry gains one (1) energy if the entry matches the expectations of one (1) or more collaborators and is taken from the tokens of those who voted to accept the entry. Another possibility is that a user who proposed on entry loses one (1) energy if the entry does not match the expectations of one (1) or more collaborators and is distributed to those who have to report the content. Following this process, energy (the CoWaBoo protocol currency) is distributed over the latter. This set of rules is an integral part of the application and can be revisited by its users.

These early reputation rules presented above, affecting reputation of terms and users, are not to be considered as compete reputation systems. They are a clear indication that the CoWaBoo protocol offers its users an accessible possibility to experiment and deploy such rules a in a concrete application. By lowering the barrier of users accessing such features we aim to make the social semantic more user oriented and trigger considerations about the competencies that application developers need to develop.

VI. Conclusions and Further Work

S2w does not imply a high level of “automation of the meaning” with formal ontologies processed by automated inferences, but focuses in situations where a semantic need is translated with the technology. Human beings need to stay in the loop, interacting during the whole lifecycle of applications, for both cognitive and cooperative reasons [13]. S2w is taking a new turn with the rise of P2P (peer to peer) storage protocols and the open public blockchain realm. Users need to come up with new analysis and development tool and face in a more learnable way web platform, protocols, applications. The semantic normativity in platforms and applications around us is not designed, with its users. However, it is influenced by the data the users themselves produce, while using them.

This paper tried to add more details and use cases on how the CoWaBoo protocol aims to reverse our habits to consume prescribed information and describe how user-oriented processes could take place [1]. We used the previous experience of social semantic web efforts to understand how this can happen and brought user driven applications as concrete examples to study this. We focused on the importance of humanly created semantics in web application development, as a means to bridge the gap between opaque algorithmic platforms and user designed, semantically rich, data representations. The GraphToLearn reputation system, appearing in the previous section, is a clear indication that complex socio-technical mechanisms do not need to be opaque or incomprehensible.

During our activities with our university students, we were surprised to notice their leapfrogging possibility to confront with this novel application building. Expressing ideas, documenting their evolution and coding them becomes possible with limited effort. The non-efficient creation process of these applications, equally, surprised us: a lot needs to be done to transform users into s2w actors. In addition, it became clear to us that the social semantic web elements in application building are part of the possibilities provided by the CoWaBoo protocol. Elements such as custom and dynamic term search based on user participation and entries reputation, implementation of group participation rules, individual or group entries’ annotation are part of these possibilities. Several problems encountered in ranking, documentation and automated search can be handled by a more detailed semantic conceptualization of these applications. This could lead us to an era where the s2w includes not only taxonomies and ontologies but real life, bottom up applications, using powerful user designed validation systems.

Although we can give not a binary answer as to whether the CoWaBoo protocol and applications provide us with a semantic and representational tool, specifically, in a context of collaboration and knowledge production, we have demonstrated how such a process could be initiated. Further work on the use of the protocol and its applications is scheduled. This work includes the multiplication of similar use cases, during the spring semester 2019, within a targeted course in the Information Systems Department (HEG) of the University of Applied Sciences in Geneva. This involves reusing the protocol through its use cases and functions, experimenting on:

- More complex group rules and valorization of transactions between participants in various groups. This should include testing of the default rules of the protocol and evaluation of the transactions functions in the application, while leading to group rules and results editable and possible to change from everyone, as long as her/his entry (definition) is voted in a group through the CoWaBoo currency.
- An alternative search experience based on GraphtoLearn and a more educated understanding of community resources as a reference to information search. Our goal is to stimulate competences as penetrating intelligence, keen perception and sound judgment in community driven curation.
- Creating new applications, scenarios and early implementations, based on the CoWaBoo protocol API (including both IPFS and Stellar protocols). These scenarios can be deployed using the protocol and its rules, or being inspired by it.

Finally, we believe that understanding web applications as potential open and descriptive protocols is a crucial step towards more transparency, less opacity, in our digital era. We intend to continue our research both as a way to unmask current opacity in digital technologies and experiment on new tools that could support collaborative and critical competencies.

References

[1] A. Priftis and J-P.Trabichet, “CoWaBoo: a descriptive protocol of learning driven applications,” The Ninth International Conference on Mobile, Hybrid, and On-Line Learning (eLmL 2017), IARIA, March 2017, pp 94-100 ISSN: 2308-4367, ISBN: 978-1-61208-541-8

[2] E. Hargittai and A. Hinnant, “Digital inequality,” Communication Research, vol. 35, no.5, p.602, 2008

[3] S. Barocas, “Data mining and the discourse on discrimination,” The Data Ethics Workshop, Conference on Knowledge Discovery and Data Mining (KDD 2014), KDD, Aug 2014. [Online]. Available from: URL: https://dataethics.github.io/proceedings/DataMiningandtheDiscourseOnDiscrimination.pdf

[4] E. Fisher and M. Naumer, “Information grounds: Theoretical basis and empirical findings on information flow in social settings,” in New directions in human information behavior, A. Spink & C. Cole, Eds. Amsterdam: Kluwer, pp. 93-111, 2006.

[5] J. Burrell, “How the machine ‘thinks’: Understanding opacity in machine learning algorithms,” SAGE Publications, pp. 23-30, 2016.

[6] K. Weller, “Knowledge Representation in the Social Semantic Web,” Knowledge Information, Berlin : De Gruyter, vol. 3, pp. 82-93, 2010.

[7] S. Cicero - Why Platforms need to be Engines of Learning — Stories of Platform Design. [Online]. Available from: https://stories.platformdesigntoolkit.com/platforms-are-engines-of-learning-4f7b70249177#.ygl8vmabw44A

[8] F. Coffield, D. Moseley, E. Hall, and K. Ecclestone, “Should we be using learning styles? What research has to say to practice,” London: Learning and Skills Research Centre, 2004.

[9] V. Lehdonvirta, & A. Robleh, A.. “Governance and Regulation.” in M. Walport (ed.), Distributed Ledger Technology: Beyond Blockchain. London: UK Government Office for Science, pp. 40-45, 2016.

[10] V. Buterin, V. Applications. Ethereum White Paper (2013). GitHub. [Online]. Available from: https://github.com/ethereum/wiki/wiki/White-Paper#applications

[11] A. Galloway, Protocol: How Control Exists After Decentralization, MIT Press, 2004.

[12] Y. Marketakis, N. Minadakis, H. Kondylakis, K. Konsolaki, G. Samaritakis, M. Theodoridou, G. Flouris, M. Doerr, “X3ML Mapping Framework for Information Integration in Cultural Heritage and Beyond,” Int. Journal on Digital Libraries, pp 1-19, 2016, dOI 10.1007/s00799-016-0179-1

[13] J-P. Cahier, M. Zacklad, A. Bénel, H. Zaher, C. Lejeune, et al. “Hy-pertopic : une métasémiotique et un protocole pour le Web socio-sémantique“. 18eme journées fran-cophones d’ingénierie des connaissances IC2007, July 2007, pp.217-228, 2007.

[14] T. Gruber, T. Gruber, "Ontology of folksonomy: A mashup of apples and oranges," Intern. Journal on Semantic Web and Information Systems, vol. 3, no. 1, pp. 1-11, 2007.

[15] C. Cattuto, D. Benz, A. Hotho, G. Stumme, J. Baumeister, M. Atzmüller, "Semantic analysis of tag similarity measures in collaborative tagging systems," LWA volume 448 of Technical Report, Department of Computer Science, University of Würzburg, pp. 18-26, 2008.

[16] J. Liu, D. M. Gruen, J. M. Bradshaw, H. Lieberman, S. Staab, "Between ontology and folksonomy: a study of collaborative and implicit ontology evolution," in IUI, ACM, pp. 361-364, 2008.

[17] R. Baeza-Yates and B. Ribeiro-Neto, “Modern Information Retrieval,” Boston, MA:Addison-Wesley Longman Publishing. Co, Inc., 1999.

[18] The SSWAP HTTP API documentation. [Online]. Available from URL: http://sswap.info/api/Home.html

[19] S. A. Goff et al., “The iPlant Collaborative: Cyberinfrastructure for Plant Biology”, Frontiers in Plant Science 2, pp. 1-16, 2011, doi: 10.3389/fpls.2011.00034

[20] C. Roth, “Socio-Semantic Frameworks. Advances in Complex Sysems,” in World Scientific, pp.135, 2013.

[21] P. Morville, Ambient Findability: What We Find Changes Who We Become, O'Reilly Media, Inc. 2005.

[22] S.D. Roy and W. Zeng, Social Multimedia Signals, Springer, Cham, 2015.

[23] N. Szabo, N. “Formalizing and Securing Relationships on Public Networks”. First Monday, vol. 2, no. 9, 1997, doi: http://dx.doi.org/10.5210/fm.v2i9.548

Resumption of Runtime Verification Monitors: Method, Approach and Application

Christian Drabek, Gereon Weiss

Fraunhofer ESK
Munich, Germany
e-mails: {christian.drabek,gereon.weiss}@esk.fraunhofer.de

Bernhard Bauer

Department of Computer Science
University of Augsburg, Germany
e-mail: bauer@informatik.uni-augsburg.de

Abstract—**Runtime verification checks if the behavior of a system under observation in a certain run satisfies a given correctness property. While a positive description of the system's behavior is often available from specification, it contains no information for the monitor how it should continue in case the system deviates from this behavior. If the monitor does not resume its operation in the right way, test coverage will be unnecessarily low or further observations are misclassified. To close this gap, we present a new method for extending state-based runtime monitors in an automated way, called resumption. Therefore, this paper examines how runtime verification monitors based on a positive behavior description can be resumed to find all detectable deviations instead of reporting only invalid traces. Moreover, we examine when resumption can be applied successfully and we present alternative resumption algorithms. Using an evaluation framework, their precision and recall for detecting different kinds of deviations are compared. While the algorithm seeking expected behavior for resumption works very well in all evaluated cases, the framework can also be used to find the best suited resumption extension for a specific application scenario. Further, two real world application scenarios are introduced where resumption has been successfully applied.**

Keywords–resumption; runtime verification; monitor; state machine; current state uncertainty; networked embedded systems; model-based.

I. Introduction

This paper extends, updates, and provides more detail on earlier research results presented at the International Conference on Trends and Advances in Software Engineering [1].

In various application areas, new kinds of services are created by combining a multitude of different software functions. Off-the-shelf products provide means to connect software functions on a physical and logical level, regardless if the functions are spread over several devices or share a common platform. However, verifying the correct functionality and identifying deviating services remains a challenge, since not only static interfaces have to be compatible but also the interaction behavior [2]. Moreover, the verification process of the final product remains incomplete, as the entire verification of embedded programs is unsolvable in general [3]. Thus, diverse approaches suggest monitoring such a system at runtime to check that it adheres to its specification [4][5][6].

A robust system continues its work after a non-critical failure or deviation from its specification occurs. Hence, it may deviate multiple times during a single execution and all deviations should be identified by the monitor. By this, the development time and effort needed to observe deviations can be reduced; especially, if they are rare and hard to be reproduced. The effort for creating such a monitor can be reduced, if artifacts from the specification phase can be reused [7]. State machines are a common way to specify interactions and protocols. However, these state machines are often limited to expected behavior and have an incomplete transition function. This means, it remains undefined what happens if an unpredicted deviation in the interaction behavior occurs. If the monitor has to terminate on a deviation, any further deviation that would be observable will be missed.

This work presents a novel approach for detecting all differences between an execution of a system and its specification using a single runtime verification monitor. Our main research goal is to enable a monitor to identify all detectable deviations instead of reporting only invalid traces. Moreover, we strive for eliminating the need to split a specification into independent properties. Therefore, we examine how the same monitor instance can resume its observation and find multiple deviations. We call this approach *resumption*.

Using resumption, the same model can be used to define valid behavior in the specification and to verify its implementation, i.e., no separate verification model needs to be created. If available, we suggest to use a reference model of a specification as basis for the monitor. Thereby, it is easier to understand deviations, as they can be directly related to the context of the whole specification. Further, the reuse of the specification guarantees compliance of the respective monitor. We examine the conditions that allow deviations to be identified and the current state uncertainty to be reduced, i.e., when resumption can succeed. We present alternative resumption algorithms and the evaluation framework used to compare them. By selecting a different resumption algorithm, the monitor can be optimized for a particular application scenario. We introduce two real world application scenarios where resumption has been successfully applied.

The rest of this paper is structured as follows. Section II introduces runtime verification using a specification-based monitor and the problem of detecting all deviations. Section III gives a survey of state-of-the-art methods to detect multiple deviations in a system's execution. The method of resumption is introduced in Section IV: First we examine unexpected behavior, before we discuss the detection of deviations and introduce the algorithms considered in this paper. Section V presents the evaluation and discusses the findings. In Section VI, we demonstrate our approach with real world applications. Section VII concludes the paper and gives an outlook on future work.

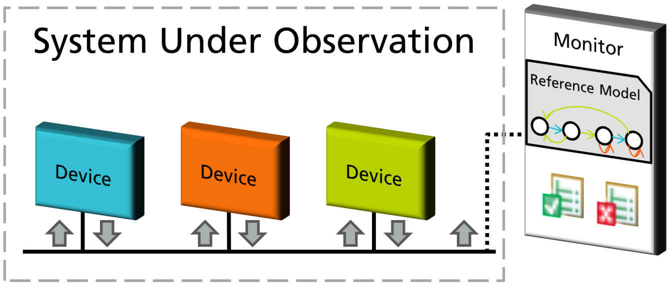


Figure 1. Monitor using a reference model to verify communication behavior of a system under observation.

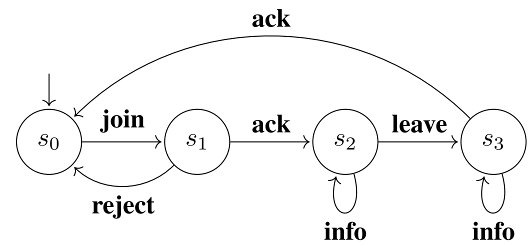


Figure 2. Example illustrating a state machine used to describe the valid communication behavior of a subscription service.

II. PROBLEM DESCRIPTION

This work considers the problem of finding all differences between an execution of a *system under observation* (SUO) and its specification using a single monitor. The core of a monitor is an analyzer which is created from the requirements. Different languages can be used to specify the analyzer [6]. In the literature, several approaches can be found, e.g., linear temporal logic [8]. Such a description can also be given as a set of states and a set of transitions between the states [9], i.e., a (finite) state machine. Further, in automotive and other embedded system domains, state machines are often used for specifying communication protocols or component interactions. We suggest using them in the form of so-called *reference models* [7], which focus on capturing valid behavior and include only critical or exemplary deviations. Such reference models can be learned from observed behavior or generated from other specification artifacts and are quite versatile. They can be used as reference for development, but may also serve as basis for a restbus simulation, or the generation of test cases. Further, a passive reference model can be used as a monitor [7]. Any reference model can be passivated by transforming actions into triggers and introducing intermediate states. It is run in parallel to the SUO and cross-checks the observed interactions with its own modeled transitions (cf. Figure 1). The communication can use a hardware bus, separate links, a middleware or other means. However, we assume the monitor taps into a (virtual) communication bus at a single point and observes the messages in order. Otherwise, this may require additional efforts, e.g., to synchronize times and merge traces, which is beyond the scope of this work. For concurrent behavior, all possible orders are expected to be modeled. The monitor uses the reference model to check the communication and produces verdicts accordingly. As this model is directly derived from the specification, the monitor effectively compares observations with the specification.

A reference-model consists of three main layers: *structual interface*, *mapping to events* and *behavior description*. Structural interface specifications define available messages and their parameters. A mapping defines constraints on the parameters. Thereby, each message can be labeled with a semantic event. The semantic event also captures the sender and receiver of the message. The set of semantic events is used to distinguish the different interactions of the SUO relevant for the specification. At runtime, there are various ways to extract the semantic events by preprocessing and slicing the observed interactions, e.g., [9][7][10][11]. In the following, we will refer to them in general as *events*.

They are used to specify the expected behavior of the SUO as a state machine $\mathrm{SM} = \langle \mathbb{E}, \mathbb{S}, \delta \rangle$.

- $\mathbb{E}$ is the set of events that can be observed.
- $\mathbb{S}$ is the set of states of the state machine, including the initial state s_0.
- $\delta \subseteq \mathbb{S} \times \mathbb{E} \to \mathbb{S}$ is the relation of transitions. It is incomplete, as unexpected behavior is omitted.
- Its size is defined as $|\mathrm{SM}| = |\mathbb{S}| + |\delta|$.

An example of such a state machine is shown in Figure 2. Sender and receiver of the events have been omitted for clarity. Where applicable, we use e_i and s_i to refer to events and states as known by the state machine; respectively, a_i and q_i are used for events and states in the sequence of execution by the SUO. For brevity, the labeling function $e(a_i)$ is omitted and a_i is used directly. Therefore, a trace is a sequence of events $a_1a_2 \ldots a_n = v \in \mathbb{E}^*$. Moreover, δ is extended to accept traces (1) and sets of states (2).

$$\delta(q_1, v) = \delta(\delta(q_1, a_1 \ldots a_{n-1}), a_n) = \delta(q_n, a_n) = q_{n+1} \quad (1)$$

$$\delta(Q, v) = \{s \in \mathbb{S} \mid \exists q \in Q : \delta(q, v) \mapsto s\} \quad (2)$$

To facilitate referral to events and states with a defined mapping in δ, let $\mathrm{dom}(\delta)$ be the domain of the partial function δ, i.e., the set of elements with a defined mapping. Further, $\mathbb{E}^s$ is the set of events with a defined transition in state s (3) and $\mathbb{S}^e$ is the set of states with a defined transition for event e (4).

$$\mathbb{E}^s = \{e \in \mathbb{E} \mid \langle e, s\rangle \in \mathrm{dom}(\delta)\} \quad (3)$$

$$\mathbb{S}^e = \{s \in \mathbb{S} \mid \langle e, s\rangle \in \mathrm{dom}(\delta)\} \quad (4)$$

A trace w is *valid*, if it describes a path through the state machine SM starting at the initial state s_0. Any subsequence of a valid trace is *expected behavior*, i.e., at least one prefix v_p and suffix v_s exist, such that $v_p v v_s = w$. In other words, expected behavior v is contained in a valid trace w and, therefore, v is a sequence of events that can be observed by following a path in the state machine. If the behavior is *unexpected*, it contains at least one event a_j with $\langle q_j, a_j\rangle \notin \mathrm{dom}(\delta)$. An *invalid* trace $\bar{w}$ contains unexpected behaviors and, thereby, contradicts the specification. The SUO *conforms* to the specification if it is in $s \in \mathbb{S}$ and emits $e \in \mathbb{E}^s$. Otherwise, it *deviates* and violates the specification. In the example from Figure 2, any sequence of events obtainable by starting at s_0 and following a path through the machine is a valid trace, e.g., *join,ack,info,info,leave,ack*. Any subsequence of this is expected behavior. An invalid trace has no path through the machine, e.g., if *info* is appended to the previous example, the trace becomes invalid and any subsequence containing *leave,ack,info* is unexpected behavior.

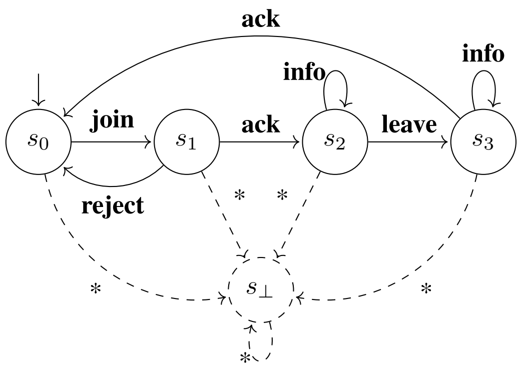


Figure 3. State machine from Figure 2 extended with transitions added for monitoring of any violation without resumption.

A monitor M $= \langle \mathbb{E}, \mathbb{S}, \delta, \mathbb{D}, \gamma \rangle$ defines specific outputs, i.e., the verdicts, for each transition of the state machine that reflect if a violation was detected. In addition to the elements of a state machine, it contains

- a verdict domain $\mathbb{D}$,
- and a verdict function $\gamma : \mathbb{S} \times \mathbb{E} \rightarrow \mathbb{D}$.

Usually, it reports only one of the following violation kinds: *invalid traces*, *unexpected behaviors* or *deviations*. Therefore, the verdict domain contains at least an accepting verdict ($\top$) and a rejecting verdict ($\bot$). To produce a verdict for a set of states Q, we assume that verdicts can be combined by an operation $\oplus$. This results in the extended verdict function given by (5).

$$\gamma(Q, a_i) = \bigoplus_{q \in Q} \gamma(q, a_i) \tag{5}$$

We call a monitor that uses $\mathbb{E}$, $\mathbb{S}$ and δ of the reference model a *specification-based monitor*. Because only expected behavior is given in the reference model, each transition corresponds to an expected observation. However, if unexpected behavior is observed, the current state of the SUO becomes undefined by definition. If the deviation was not critical and the SUO is implemented in a resilient way, the SUO will often be able to continue. The SM in Figure 2 shows a SM that specifies the communication behavior related to a subscription service. It has only accepting transitions; a possible resolution of implicit transitions is shown in Figure 3. The wild-card '*' matches any unbound event. Exactly all violations are unbound in the original state machine. Hence, the extended state machine enters the rejecting state $s_\bot$ after a violation. However, as the state machine remains there, the respective monitor only detects invalid traces by a first deviation. To detect further deviations or unexpected behaviors, the monitor must be able to resume its observation.

III. Related Work

Various areas address the problem of detecting differences between a system's behavior and its specification model. This section gives a brief overview of how existing approaches match specified model and observed behavior.

Conformance checking compares an existing process model with event logs of the same process to uncover where the real process deviates from the modeled process [12]. It is used offline, i.e., after the SUO finished its execution, because the employed data mining techniques to match model and execution are computationally intensive. They can only be used efficiently once the complete logs are available. Cook et al. [13] use a best-first search to find the necessary insertions and deletions of events to transform the given event stream into one that exactly matches the model. Based on the *least changes* needed, they give a measure for the difference of an observed trace from its process description. Reger [14] suggests including the origin of an event into the analysis to find sensible edits that are consistent for the same origin and correct the trace. He transforms trace, edits and state machine into weighted transducers T_T, T_X and T_S. Transducers are composable algorithmic transformations, i.e., automations that read an input sequence and write the result of the transformation as an output sequence. Each step of a weighted transducer is associated with a cost. A three-way composition [15] of the three transducers can be performed in $O(|T_X|\,|T_T|)$ without a large intermediate result. Every path in the composition represents a way to edit the trace to match a path in the state machine. This composed transducer can then be searched for the lowest overall cost edit. The computational complexity to find this edit sequence requires it to be executed offline. In contrast, resumption does not yield precise edits, but can be performed online, in parallel to the execution of the SUO.

Allauzen and Mohri [16] have shown that such a composed transducer can be computed in linear-space. However, the computation is still expensive with regard to time. Therefore, Cook and Wolf [17] utilize *pruning* to reduce the cost of finding a low-cost goal by discarding portions of the search space that look unpromising. They reason that even though pruning breaks the guarantee of identifying the lowest-cost goal, it often has negligible effects on the result while reducing the search space dramatically. They suggest using *cost pruning* and *position pruning*. The former eliminates nodes that are estimated to result in a higher than expected overall cost; the latter removes nodes that are τ_{prune} steps in the trace behind the current best node from the search. In contrast to these mining algorithms, the verification with resumption presented in this paper works at runtime. However, we would like to note that, if position pruning is used, all information to perform the search for an edit path is available at runtime with an offset of τ_{prune} steps. Therefore, we chose the *least changes* approach for comparison with resumption.

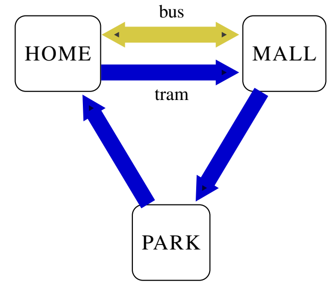
Model-based testing aims to find differences between the behavior of a system under test (SUT) and a valid behavior model [18]. An environmental [19] or embedded [20] test context stimulates the SUT with test sets, i.e., selected input sequences. The SUT's outputs are then compared with the expected output from the behavior model. *Homing sequences* actively maneuver the SUT into a known state [21]. Generally, these sequences reduce the current state uncertainty by utilizing separating or merging sequences [22]. Former assure different outputs for two states, latter move the machine into the same state for a given set of initial states. Minimized Mealy Machines are guaranteed to have a homing sequence [22]. However, a passive monitor cannot influence the SUO. Therefore, it cannot actively force the system to a known state. Nevertheless, we utilize occurrences of separating and merging sequences to reduce the number of possible candidates for the current state.

In general, *runtime verification* can be seen as a form of passive testing with a monitor, which checks if a certain run of a SUO satisfies or violates a correctness property [5]. Such a dynamic analysis is often incomplete, i.e., it may yield false negatives; however, this incompleteness helps neutralize limitations of static analysis [9], e.g., state-explosion. The observation of communication is well suited for black box systems, as no details about the inner states of the SUO are needed. Further, the influence on the SUO by the test system is reduced by limiting the intrusion to observation. In case the deviations are solely gaps in the observation, a Hidden Markov Model can be used to perform runtime verification with state estimation [23]. Runtime verification frameworks, such as TRACEMATCHES [10] or JAVAMOP [11], preprocess and filter the input before it is passed to a monitor instance. Thereby, each monitor only observes relevant events. These stages implement the first two layers of a reference model [7]. The monitor uses the reference model's third layer to answers *yes* or *no* to the question, if the provided trace fulfills or violates the monitored property. This is also the case, if it contains multiple violations. However, it may be of interest to identify all violations, similar to conformance checking. Simply keeping the monitor running after it encountered and reported a violation only works in very specific scenarios. This is similar to using the resumption algorithm *Waiting* presented in Section IV-D. Nevertheless, if the properties are carefully chosen, multiple instances of the monitor can match different slices of an input trace [10][24]. Such properties can be extracted from the behavior model [3] or by data mining techniques from a running system or traces [17][21][25][26][27]. However, the former implies additional design work and the latter requires a correctly working system to learn from. Furthermore, this creates a secondary specification that needs to be maintained and validated. In contrast, the presented resumption enables reuse of an available specification by automatically augmenting it with robustness for verification.

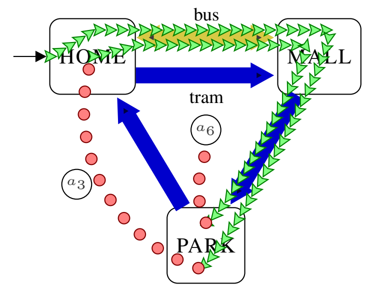
IV. RESUMPTION

A specification-based monitor, such as shown in Figure 3, will only be able to find the first deviation from a specification, since it enters the final state $s_\perp$ at this point. Different techniques can be applied in order to create robust monitors and to find deviations beyond the first. Up to now, this is usually done manually and requires additional design work, e.g., to repeatedly add transitions and triggers or to artificially split the specification into multiple properties that can be checked separately. Therefore, we suggest using a generic definition for how a monitor can resume its duty. This section describes *resumption*, a method that enables a monitor to analyze a trace for more than one violation with respect to the same property.

For illustration, runtime verification can be seen as a game for two players: DECEIVER and VERIFIER. DECEIVER acts as SUO and he covertly moves on a map, which represents SM, while leaving a trace of moves. After each move, VERIFIER, the monitor, has to tell if DECEIVER violated the specification given by the map. She knows only the trace and the map. Figure 4a shows an example map for the game. Figure 4b depicts the path that DECEIVER has chosen in this example, producing a trace *bus,tram,bus,bus,tram,bus*. He starts at HOME. On step a_3 and a_6, he choses to *deviate* and claims to move using *bus*, actually not available at state PARK. For a_3, he decides to go HOME; for a_6, he may still jump to any state. All other moves *conform* to the map.



(a) Example map known by DECEIVER and VERIFIER.



(b) DECEIVER's hidden movement. Arrows show conforming, circles deviating moves.

Figure 4. Example illustrating a game of runtime verification, where VERIFIER (monitor) tries to detect, if DECEIVER (SUO) violates the specification given as a map (state machine).

Knowing the current state of DECEIVER it is trivial for VERIFIER to follow the conforming movement of DECEIVER using only trace and map. She can infer his next states using the moves recorded in the trace and δ. Further, she can easily identify his first deviation. Then, DECEIVER's move is not contained in δ. This exemplifies how monitors without resumption work.

Theorem 1: VERIFIER can detect deviations using the verdict function given in (6), if she knows DECEIVER's current state is q_i.

$$\gamma(q_i, a_i) \mapsto \begin{cases} \top, & \text{if } \langle q_i, a_i \rangle \in \text{dom}(\delta) \\ \perp, & \text{otherwise.} \end{cases} \tag{6}$$

Proof: If VERIFIER knows DECEIVER is in state q_i and the event is a_i, she can look up his next state in δ. If a_i is available at q_i, i.e., $\langle q_i, a_i \rangle \in \text{dom}(\delta)$, DECEIVER cannot deviate using a_i. If a_i is not available at q_i, he has to deviate to be able to use it. Thereby, if VERIFIER knows the current state of DECEIVER, she can directly tell if he deviates. ■

However, after a deviation, VERIFIER does not know to which state DECEIVER has moved. She is uncertain of his current state. Becoming aware of this uncertainty and reducing it in order to be able to resume runtime verification is *resumption*. To express multiple deviations in a sequence of verdicts, each verdict refers to the trace after the last reported violation. First, we show that detection of all unexpected behavior is possible through resumption. Later, we demonstrate how resumption can be employed to find minimal subsequences of a trace, each containing exactly one detectable deviation.

A. Resumption for Unexpected Behavior Detection

Unexpected behavior is defined as any sequence of events the SUO may not emit according to the specification. Behavior is unexpected if it cannot be matched to anywhere in the specification. We assume it is unknown how the behavior of the SUO's components is implemented internally and only their stimuli and responses are observable (black-box assumption). We expect these interactions in the SUO to be specified by a state machine SM. If the observed events can be matched to SM, such behavior is assumed to conform to the specification. Otherwise, i.e., in case of a violation, the new state of the SUO is unknown as the observations are the only available information. In general, no restrictions on the new state can be given, i.e., such a transition may even be non-deterministic. A monitor M knows only the specification and observes the events emitted by the interactions in the SUO.

More formally, unexpected behavior is a trace that contains an event a_j with $\langle q_j, a_j\rangle \notin \mathrm{dom}(\delta)$. As q_j is unknown to VERIFIER, it is replaced by $Q_j^k = \delta(\mathbb{S}, a_k \dots a_{j-1})$, the set of states reachable with expected behavior from any state with the trace starting at step k and ending before step j. Please note that $Q_j^j = \mathbb{S}$.

Theorem 2: A trace can be split within space $O(|\mathrm{SM}|)$ and time $O(|\mathbb{S}|)$ per step so that each segment contains exactly one unexpected behavior. For each appended event, it can be decided in this space and time if it belongs to the same or a new segment.

Proof: For brevity, we define $\rho_{i+1} = \rho_i'$. A valid segmentation of the trace is a sequence of indexes determined by (7). Each of the segments is delimited by two indexes ρ_i and ρ_i'. The trace $v_i = a_{\rho_i} \dots a_{\rho_i'}$ always encloses exactly one unexpected behavior. If there was no unexpected behavior in v_i, then $Q_{\rho_i'}^{\rho_i}$ would not be empty. If there was additional unexpected behavior at step ρ with $\rho_i < \rho < \rho_i'$, then $Q_\rho^{\rho_i}$ would be empty already and ρ would have been chosen as minimum. Therefore, exactly one unexpected behavior is contained in v_i.

$$\rho_0 = 1 \wedge \rho_i' = \min\{\rho \mid \rho > \rho_i \wedge Q_\rho^{\rho_i} = \emptyset\} \tag{7}$$

For each step, the set of successors of the possibly active states must be calculated, i.e., $\delta(Q_{\rho_i'}^{\rho_i}, a_i)$. This requires up to $|\mathbb{S}|$ lookups in the transition relation of size $|\delta|$. Each lookup takes time $O(1)$, if perfect hashing is used. Therefore, each step needs time $O(|\mathbb{S}|)$ and space $O(|\mathbb{S}| + |\delta|) = O(|\mathrm{SM}|)$. ■

To relate this to resumption, we explain how VERIFIER checks DECEIVER's movement with no information about DECEIVER's initial state. VERIFIER has to find a verdict for all possible locations of DECEIVER. The transition function δ is already extended by (2) to accept a set of states Q and to return all states reachable with event e from a state in Q. δ will return an empty set if none of the possible states has a matching transition for the event. To continue verification, this function needs to be extended with the actual resumption. VERIFIER's uncertainty of DECEIVER's state resets if she cannot confirm his movement. The most general assumption is that DECEIVER is in any state. This is reflected in transition function δ^+ (8).

$$\delta^+(Q, e) = \begin{cases} \delta(Q, e), & \text{if } \gamma\text{(Q,e) = } \top \\ S, & \text{otherwise.} \end{cases} \tag{8}$$

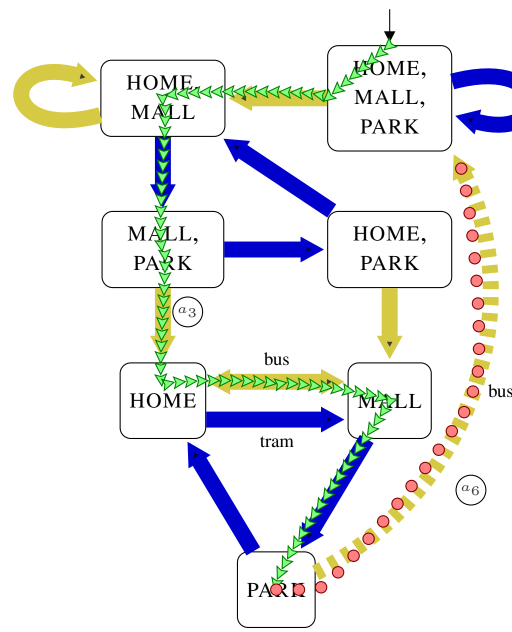


Figure 5. Power map of the example from Figure 4a including the interpretation of DECEIVER's trace for unexpected behavior from VERIFIER's perspective.

Similar, the verdict function γ is extended by (5) to produce a verdict for multiple states using a combination operation $\oplus$. For detecting unexpected behavior, this may be done using $\oplus_{eb}$ (9), where Γ is the set of verdicts to merge. As long as Q^e is not empty, there is a state in Q with an accepting transition for e and the behavior remains expected. Otherwise, unexpected behavior is detected and the next segment of the trace starts.

$$\oplus_{eb}(\Gamma) = \begin{cases} \top, & \text{if } \top \in \Gamma \\ \bot, & \text{otherwise} \end{cases} \tag{9}$$

Instead of analyzing the model at runtime, power set construction can be used to resolve the non-determinism beforehand. Also, this allows to illustrate resumption for the game example. DECEIVER's moves are translated to the *power map* $\mathrm{SM}^{\mathcal{P}} = \langle \mathbb{E}, \mathcal{P}(\mathbb{S}), \delta^{\mathcal{P}} \rangle$, starting at state $s_{\mathbb{S}}$. $\mathcal{P}(\mathbb{S})$ is the set of all subsets, the power set, of $\mathbb{S}$. $\delta^{\mathcal{P}}$ is δ^+, except that it returns the respective state for the set of states instead of the set itself. Transitions that were unexpected behavior in SM are routed to $s_{\mathbb{S}}$ in $\mathrm{SM}^{\mathcal{P}}$. $\delta^{\mathcal{P}}$ is complete and deterministic. Thereby, VERIFIER always knows DECEIVER's precise state in $\mathrm{SM}^{\mathcal{P}}$ and merely has to follow his movement (see Theorem 1). She starts assuming DECEIVER is in any state of the original map, i.e., in $s_{\mathbb{S}}$ (cf. Figure 5). After step a_3, the states DECEIVER could be in using only expected behavior are narrowed down to HOME. As we know from Figure 4b, DECEIVER deviated at a_3 to HOME. However, this is still expected behavior as there would be no deviation if DECEIVER started at MALL. This is shown in Figure 6. The second deviation at a_6 is unexpected behavior, as there are no other valid options left. Compared to Theorem 2 above, checking for unexpected behavior in $\mathrm{SM}^{\mathcal{P}}$ only takes time $O(1)$ for each step; however, the extended map requires significantly more space: $O(2^{|\mathbb{S}|})$.

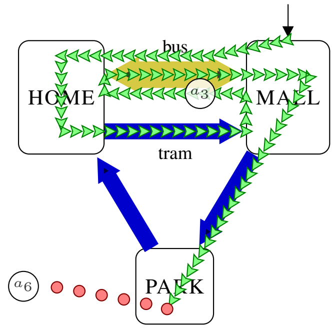


Figure 6. Alternative of DECEIVER's hidden movement producing the same trace as Figure 4b, but with no deviation at a_3 by starting at a different state.

B. *Resumption for Deviation Detection*

The previous section revealed that it is possible for a monitor to detect all unexpected behaviors of a SUO. This section examines how resumption can help to detect deviations of a SUO. A deviation is an event that is not allowed in the SUO's current state. Therefore, deviations are directly linked to the SUO's state. In terms of the example, VERIFIER must detect if DECEIVER chose to deviate with a move. This changes how VERIFIER can handle uncertainty about DECEIVER's state. She may be unable to decide if DECEIVER deviated. This requires a third verdict to express that she is *inconclusive* ($\top\!\!\!\bot$).

Theorem 3: VERIFIER is exactly then conclusive about DECEIVER's move a_i, if the move is defined or undefined for all states Q she knows he could be in (10).

$$\gamma(Q, a_i) = \top\!\!\!\bot \iff \exists s_1, s_2 \in Q : \gamma(s_1, a_i) \neq \gamma(s_2, a_i) \quad (10)$$

Proof: The implication from left to right follows from the limitations of DECEIVER's moves. If none of the states in Q has a transition for the event, DECEIVER cannot conform; therefore, he *obviously deviates*. If all of the states in Q have a transition for the event, DECEIVER cannot deviate; therefore, he *obviously conforms*. In all other cases, VERIFIER cannot give a conclusive verdict, as there is at least a state s_1 in Q that has a transition for event a_i and a state s_2 that has not. If her verdict was $\bot$, DECEIVER may in fact have been in s_1 and actually conforms; if it was $\top$, he may have been in s_2 and deviates. Thereby, her verdict must be $\top\!\!\!\bot$. If the verdict is the same for all individual states in Q, it must be $\top$ or $\bot$ and it follows the implication from right to left. ∎

This helps to formulate $\oplus_d$ (11) for combining verdicts. Identical verdicts combine to the same verdict and different verdicts combine to an inconclusive verdict.

$$\oplus_d(\Gamma) = \begin{cases} \top, & \text{if } \{\top\} \equiv \Gamma \\ \bot, & \text{if } \{\bot\} \equiv \Gamma \\ \top\!\!\!\bot, & \text{otherwise} \end{cases} \quad (11)$$

This sounds promising, as at least in some cases there can be conclusive verdicts. Further, one might expect that VERIFIER should be able to close in on DECEIVER's position eventually. However, as it is always possible that DECEIVER deviates, Q_{i+1} must be the set of all states, unless he obviously conforms. This is enforced by δ^+.

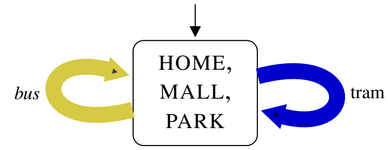


Figure 7. Power map of the example from Figure 4a for deviation detection from VERIFIER's perspective.

As a consequence, VERIFIER can only reduce her current state uncertainty, if she observes a sequence of obviously conforming events. Further, the sequence must eliminate s-tates from her current state uncertainty. In the following, we examine if such sequences exist. Sandberg [22] describes how *synchronization* and *homing* sequences can be found in completely specified, deterministic Mealy machines in time $O(|\mathbb{S}|^3 + |\mathbb{S}|^2 \cdot |\mathbb{E}|)$ — in case one exists. A synchronization sequence is a trace $x \in \mathbb{E}^*$ that transitions the machine into a single known state, regardless the initial state, i.e., $|\delta^+(S, x)| = 1$. In essence, it consists of a sequence of *merging* sequences. A homing sequence may also include *separating* sequences. It is a trace $x \in \mathbb{E}^*$ that guarantees different outputs for different target states (12). While a state machine is not guaranteed to have any synchronization sequences, e.g., the automation in Figure 4a has none, a minimized Mealy machine always has a homing sequence [22].

$$\begin{aligned} &\forall s_1, s_2 \in S : \\ &\quad \delta(s_1, x) \neq \delta(s_2, x) \implies \gamma(s_1, x) \neq \gamma(s_2, x) \end{aligned} \quad (12)$$

Homing sequences can be mapped to the deviation game. SM contains no output and is incomplete. For former, γ could act as replacement and latter can be mitigated by using δ^+. Then, we call them *unique sequences*, as they identify a unique state. However, the pure possibility of DECEIVER deviating negates the benefit of a separating sequence as this resets VERIFIER's search; i.e., she has to consider all possible states again after the possible deviation. As there is no other output available, separating sequences would work only under the assumption that no violations occur during resumption. As SM has no merging sequence, $SM^{\mathcal{P}}$ for deviation detection could be trimmed to Figure 7. VERIFIER knows DECEIVER confirms using *tram* but this does not tell anything about DECEIVER's location. DECEIVER could be in PARK; therefore, if he uses *bus*, VERIFIER is inconclusive and has to restart at $s_{\mathbb{S}}$. An option would be to resolve undefined transitions differently. However, this would impose additional assumptions on the deviations. Nevertheless, we can use the knowledge about unexpected behavior to conclude on deviations.

Theorem 4: Each trace segment $a_{\rho_i} \dots a_{\rho'_i}$ of the split trace contains at least one deviation.

Proof: For proof by contradiction, we assume that there is no deviation in the trace segment. If there is no deviation, only conforming moves were used by DECEIVER. Then, there must be at least one path in SM matching the trace, i.e., $\delta(\mathbb{S}, a_{\rho_i} \dots a_{\rho'_i}) \neq \emptyset$. Therefore, the current state uncertainty $Q^{\rho_i}_{\rho'_i}$ should contain at least one state. However, this conflicts with (7), that ensures $Q^{\rho_i}_{\rho'_i} = \emptyset$. Hence, the trace segment must contain at least one deviation. ∎

The trace segments span from one unexpected behavior to the next. Somewhere within each segment is a deviation. Yet, we want to locate the deviations as precise as possible. The segments are constructed in a way so that they always end with unexpected behavior. Therefore, they cannot be cut on that side. The following will show that their length can be limited by maximizing the start index while retaining the constraint for the trace segment, i.e., that it contains unexpected behavior.

Theorem 5: The trace $a_{\kappa_i} \dots a_{\rho_{i+1}}$ is the closest possible enclosure of a deviation before step ρ_{i+1} that retains the notion of unexpected behavior in the segment, with κ_i defined by (13).

$$\kappa_i = \max\{\kappa \mid \rho_i \leq \kappa < \rho'_i \wedge Q^{\kappa}_{\rho'_i} = \emptyset\} \tag{13}$$

Proof: The proof for existence of a deviation in the trace stays the same as for Theorem 4: $Q^{\kappa}_{\rho'_i}$ is only empty if it contains a deviation. Thus, it remains the proof that there is no shorter segment that could enclose the deviation. If $\kappa_i = \rho'_i - 1$ DECEIVER is obviously deviating with $a_{\rho'_i}$, as no state in $\mathbb{S}$ has a transition for the event. Since there must be at least one event to deviate, there cannot be a shorter sequence in this case. Equation (13) selects the greatest start index κ_i that still contains unexpected behavior and thereby a deviation. We cannot reduce the end index as it was already selected to be minimal when starting at ρ_i by (7). Starting at a later index than ρ_i can only increase the current state uncertainty, because the later starting segment must consider at least all paths from the earlier, i.e., $Q^k_j \subseteq Q^{k+1}_j$. Moreover, $Q^{\kappa_i}_{\rho'_i}$ can only be empty if $Q^{\rho_i}_{\rho'_i}$ is empty. Therefore, the closest possible enclosure is the trace $a_{\kappa_i} \dots a_{\rho_{i+1}}$. ■

κ can be calculated by searching unexpected behavior using backwards steps, starting from step ρ'_i. As following transitions backwards may yield up to $|\mathbb{S}|$ source states for the same event, the time needed for each of the $\rho'_i - \kappa_i \leq \rho'_i - \rho_i \leq |v|$ steps is increased to $O(|\mathbb{S}|^2)$ compared to what was shown for the forward search in Theorem 2. Preparing the reverse lookup tables will require space $O(|\mathbb{S}|^2 \cdot |\mathbb{E}|)$. It follows that there is at least one deviation located in the intersection of unexpected behavior found forward and backward. Moreover, the existence of κ_i implies that each segment $a_{\rho_i} \dots a_{\kappa_i - 1}$ must be expected behavior, even if it actually contains deviations. The deviations perfectly mimic expected behavior and cannot be detected. This can also be seen by the options of DECEIVER in $\mathrm{SM}^{\mathcal{P}}$.

Theorem 6: Deviations in the trace segment $a_{\rho_i} \dots a_{\kappa_i - 1}$ cannot be detected.

Proof: According to Theorem 5, segment $a_{\kappa_i} \dots a_{\rho'_i}$ contains exactly one unexpected behavior and the unexpected behavior happens at or after step κ_i. As there is only one unexpected behavior in the complete segment, there cannot be another before step κ_i. Therefore, any deviation that may have occurred in this part of the segment is observed as expected behavior and cannot be detected. ■

The deviations cannot be detected because there is no possibility to distinguish them from expected behavior with the available information. However, if additional or more detailed observations can be obtained from the system, they may become detectable.

Moreover, if the current state uncertainty is reduced to a single state without the (unobservable) occurrence of a deviation, the detected unexpected behavior is the deviation. Therefore, if given enough time between deviations, monitoring with resumption will identify exactly the deviations.

Theorem 7: If n non-overlapping unique sequences are observed without unexpected behavior, the assumed state is the actual state of the SUO, unless n or more undetected deviations occurred. However, unless one unique sequence is an obviously conforming synchronization sequence, there may have been at least n deviations.

Proof: For the first part, we show that not all unique sequences can be mislead with $n-1$ deviation. As the unique sequences do not overlap, there remains at least one unique sequence u without a deviation. As u does not contain a deviation, it reveals the actual state of DECEIVER. Theorem 1 guarantees the detection of deviations if the current state is known. Moreover, u must be the last of the unique sequences. Otherwise, the deviation would have been detected and there would be unexpected behavior. With an additional n-th deviation, DECEIVER can mislead all unique sequences, therefore, his actual state may remain uncertain. For the second part, we recall how unique sequences are constructed. Each sequence is unique, as it reduces $|Q|$ to 1. Like homing sequences, the reduction can be achieved by merging or separation sequences. A merging sequence uses the structure of the state machine, e.g., if transitions for the same event lead from two states to a single one. During a merging sequence, no deviation can occur by definition. As a synchronization sequence consists of concatenated merging sequence, it is obviously conforming and no deviations could have happened when it is observed. The observation of the synchronization sequence itself guarantees the actual state of DECEIVER is known. Separation sequences, however, work differently. They remove states from the current state uncertainty for which DECEIVER would have to deviate for the chosen event. Hence, there is at least one possibility to deviate in each separation sequence. If none of the observed unique sequences is a synchronization sequence, each contains at least one distinct separation sequence. As there are n unique sequences, there may have been at least n deviations. ■

Detecting multiple unique sequences before detecting unexpected behavior may sound unlikely. Nevertheless, sometimes rare deviations are of interest. In this case, there may be many events between deviations that can be used to resume verification. Protocols such as the subscription service in Figure 2, often contain *unique events*. Therefore, detecting multiple unique events or short unique sequences increases the confidence that you were not deceived. However, only the detection of unexpected behavior can be guaranteed.

C. Resumption Extension

Any specification-based monitor may be extended with a resumption extension. Even a monitor that has a complete transition function may be improved by resumption, if it has unrecoverable states, like $s_\perp$ in Figure 3. Sub-scripts are used to distinguish between the original monitor ($\mathcal{L}$), the extension ($\mathcal{R}$), the extended monitor ($\mathcal{E}$) and their components respectively. $\mathrm{M}_{\mathcal{E}}$ is created by combining the sets and functions of $\mathrm{M}_{\mathcal{L}}$ with $\mathrm{M}_{\mathcal{R}}$, where $\mathrm{M}_{\mathcal{L}}$ is preferred. However, $\delta_{\mathcal{R}}$ may override $\delta_{\mathcal{L}}$ for selected verdicts, e.g., $\perp$.

Example 1 (Resumption Extension): Figure 8a depicts a possible extensions of the SM in Figure 2. Instead of entering a final rejecting state for unexpected events, the extended monitor ignores the event and stays in the currently active state.

The resulting $M_{\mathcal{E}}$ has a complete transition function and is able to continue monitoring after reporting deviations. Thereby, the original monitor is extended by resumption.

While a resumption extension can be created in an arbitrary way, we suggest to use a resumption algorithm ($\mathcal{R}$) to create the extension. The algorithm's core function (14) takes an event and a set of (possible) active states as input. It returns the set of states that are candidates for resumption. The $\mathcal{R}$-based resumption extension can be easily exchanged to adjust the monitor to the current application scenario. Let $\mathbb{S}_C = \mathbb{S}_{\mathcal{L}} \cup \{s_{\mathcal{R}}\}$ and $\mathcal{P}(\mathbb{S}_C)$ be the set of all its subsets.

$$\mathcal{R} : \mathcal{P}(\mathbb{S}_C) \times \mathbb{E} \to \mathcal{P}(\mathbb{S}_C) \tag{14}$$

Using $\mathcal{R}$, the additional states and transitions needed for the extension of the original monitor can be derived. For finite sets $\mathbb{S}_{\mathcal{L}}$ and $\mathbb{E}$, a preparation step creates the states $\mathcal{P}(\mathbb{S}_C) \setminus \mathbb{S}_C$. The transitions are derived by evaluating $\mathcal{R}$ to find the target state. If $\mathcal{R}(q, e)$ returns an empty set or solely states that cannot reach any state in $\mathbb{S}_C$, it reached a finally non resumable state. The existence of such states depends on $\mathcal{R}$ and the specification. All states not reachable from a state in $\mathbb{S}_C$ can be pruned.

An alternative is using $\mathcal{R}$ at runtime as transition function during resumption. If $\mathcal{R}$ returns solely a single state in $\mathbb{S}_{\mathcal{L}}$, $M_{\mathcal{L}}$ can resume verification in that state. Otherwise, the candidates are tracked in parallel while removing non-conforming ones. If $s_{\mathcal{R}}$ is a candidate, $\mathcal{R}$ is used to transition the candidate set, otherwise $\delta_{\mathcal{L}}$. $\gamma_{\mathcal{R}}$ is created using (5) with a suitable $\oplus$.

D. Resumption Algorithms

This section introduces algorithms that can be used for resumption. These algorithms are often mimicked to extend specifications manually for creating robust monitors. Based on an observed event and a set of candidates for the active state, $\mathcal{R}$ will determine possible states of SUO with respect to the observed property. The presented algorithms can generally be categorized into *local* and *global* algorithms. The former are influenced by the state that was active before the deviation, while the latter analyze all states equally. Each of the algorithms can be used to construct a replacement for δ^+ introduced in subsection IV-A and represents a different *error model*. They are only guaranteed to find all unexpected behaviors, if all occurring deviations match this error model. However, certain assumptions can lead to a significantly simpler $M_{\mathcal{E}}$. The results of applying the algorithms on the SM from Figure 2 are shown in Figure 8.

The local algorithm *Waiting* (15) is inadvertently used when interpreting a trace with an unaltered state machine. Implementations usually ignore superfluous messages and remain in the same state, waiting for the next valid event. Therefore, the algorithm introduces no runtime overhead. It resumes verification with the next event accepted by the previously active state q, i.e., it stays in q and skips all events not in $\mathbb{E}^q$. This creates loops at every state as shown in Figure 8a for the example. However, $\mathcal{R}_{\text{wait}}$ requires that all deviations are superfluous message that may be ignored. Otherwise, the guarantee to find all unexpected behaviors will no longer hold. Therefore, it is expected to perform badly for other deviations and may stall in many scenarios.

$$\mathcal{R}_{\text{wait}}(\mathbb{S}_{in}, e) = \mathbb{S}_{in} \tag{15}$$

An obvious danger is, SUO may never emit an event that is accepted by the active state. Therefore, the next algorithms also considers states around the active state. The used distance measure $\|s_s, s_t\|$ is the number of transitions $\in \delta_{\mathcal{L}}$ in the shortest path between a source state q_s and a target state q_t. The extension $\|\mathbb{S}_s, \mathbb{S}_t\|$ is the transition count of the shortest path between any state in $\mathbb{S}_s$ and any state in $\mathbb{S}_t$. The algorithm *Nearest* (16) resumes verification with the next event that is accepted by any state reachable from the active state. Figure 8b shows the extension of the example. If multiple transitions match, it chooses the transition reachable with the fewest steps from the previously active state. A static calculation of $\mathcal{R}_{\text{near}}$ requires additional states only if tie-breakers are needed. $\mathcal{R}_{\text{near}}$ assumes that the deviations will be caused by skipped messages. It will resume on the next matched event, unless the two closest valid states require the same number of steps. As it only looks forward, superfluous or altered messages may cause it to errantly skip ahead.

$$\mathcal{R}_{\text{near}}(\mathbb{S}_{in}, e) = \underset{q_t \in \delta_{\mathcal{L}}(\mathbb{S}_C, e)}{\operatorname{argmin}} \min_{q_s \in \mathbb{S}_{in}} \|q_s, q_t\| \tag{16}$$

The algorithm *Nearest-or-Waiting* (17) is a combination of *Nearest* and *Waiting*. It measures the length of the shortest path from the active state to a state returned by *Nearest* and the shortest path of any candidate state with an accepting transition to the active state. If the latter path is shorter, *Waiting* is used. The extension of the example using $\mathcal{R}_{\text{n-o-w}}$ is depicted by Figure 8c. The idea is to ignore superfluous messages and identify them by looking as far back as was required to look forward to find a match.

$$\mathcal{R}_{\text{n-o-w}}(\mathbb{S}_{in}, e) = \begin{cases} \mathcal{R}_{\text{wait}}, & \text{if } \|\mathbb{S}_C^e, \mathbb{S}_{in}\| < \|\mathbb{S}_{in}, \mathcal{R}_{\text{near}}\| \\ \mathcal{R}_{\text{near}}, & \text{otherwise} \end{cases} \tag{17}$$

Global algorithms consider the whole specification to identify the current communication state. Therefore, they analyze all states equally to keep all options open for resumption.

Unique-Event (18) resumes verification if the event is unique, i.e., the event is used on transitions to a single state only. $\mathcal{R}_{\text{u-e}}$ is the only examined $\mathcal{R}$ that ignores all input states. As there is only one target state of a unique event in the state machine, the algorithm considers this a *synchronization point*. For other events, a resumption state ($s_{\mathcal{R}}$) is entered. This extension is illustrated for the example in Figure 8d. Therefore, a static calculation only needs a single additional state.

$$\mathcal{R}_{\text{u-e}}(\mathbb{S}_{in}, e) = \begin{cases} \delta_{\mathcal{L}}(\mathbb{S}_C, e), & \text{if } |\delta_{\mathcal{L}}(\mathbb{S}_C, e)| = 1 \\ \{s_{\mathcal{R}}\}, & \text{otherwise} \end{cases} \tag{18}$$

Unique-Sequence (19) extends the previous algorithm to unique sequences of events, as unique events may not be available or regularly observable in every specification. $\mathcal{R}_{\text{u-s}}$ follows all valid paths simultaneously and resumes verification if there remains exactly one target state for an observed sequence. Similar to homing sequences used in model-based testing, $\mathcal{R}_{\text{u-s}}$ aims to reduce the current state uncertainty with each step. It evaluates which of the input states accept the event. If the observed event is part of a separating sequence, the non-matching states are removed from the set. If a merging sequence was found, the following $\delta_{\mathcal{L}}$-step returns the same state for two input states and the number of candidates is further reduced. If there are homing sequences for $\mathcal{L}$ and the

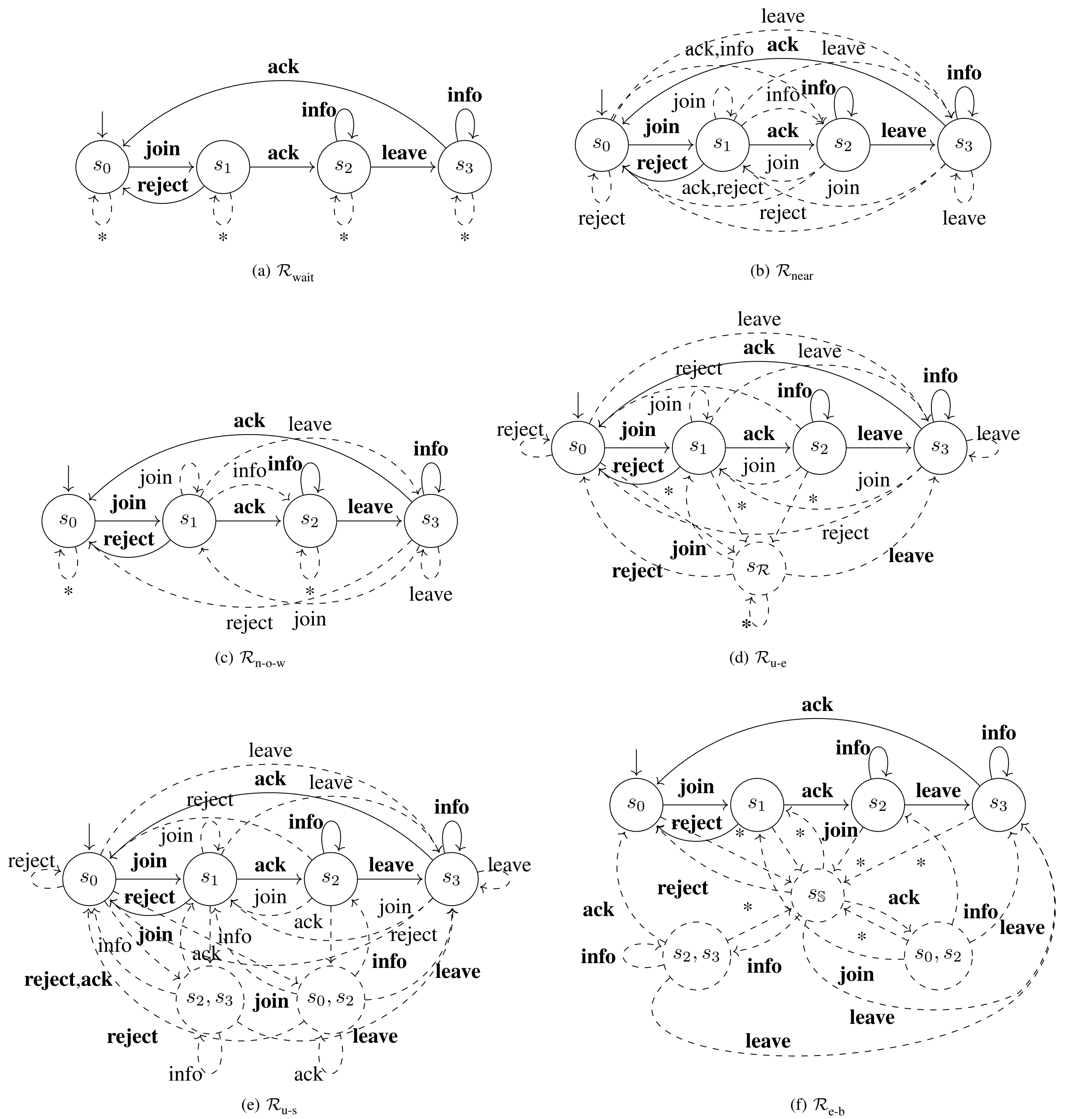


Figure 8. SMs with states and transitions added (dashed) by different $\mathcal{R}$. **Bold** labels indicate an accepting and regular labels a rejecting verdict. The wild-card '*' matches all events that have no other transition in the state.

SUO emits one, $\mathcal{R}_{\text{u-s}}$ will detect it. Any unexpected behavior causes $\delta_{\mathcal{L}}(\mathbb{S}_{in}, e)$ to be empty and therefore resets the set of possible candidates to any state accepting the event, i.e., the resumption is reset. New intermediate states are created to capture the current state uncertainty during resumption. For example, 's_2, s_3' in Figure 8e means that $\text{M}_{\mathcal{E}}$ considers the SUO in state s_2 or s_3. Therefore, the size of the static calculation is limited by $|\text{SM}^{\mathcal{P}}|$.

$$\mathcal{R}_{\text{u-s}}(\mathbb{S}_{in}, e) = \begin{cases} \delta_{\mathcal{L}}(\mathbb{S}_{in}, e), & \text{if } \delta_{\mathcal{L}}(\mathbb{S}_{in}, e) \neq \emptyset \\ \delta_{\mathcal{L}}(\mathbb{S}_{C}, e), & \text{else if } \delta_{\mathcal{L}}(\mathbb{S}_{C}, e) \neq \emptyset \\ \{s_{\mathcal{R}}\}, & \text{otherwise} \end{cases} \quad (19)$$

The formal analysis in this extended version of [1] has shown that $\mathcal{R}_{\text{u-s}}$ is not the most general case. Like the other algorithms, it can be used to extract trace segments containing deviations matching its error model. However, resumption with $\mathcal{R}_{\text{u-s}}$ uses the last event of the previous unexpected behavior. Hence, verification may be deceived as this reflects a deviation to a state accepting the event instead of any state. Therefore, we introduce the algorithm *Expected-Behavior* (20) that takes into account all expected behavior. It is the resumption algorithm version of the candidate selection δ^+ given in (8). As shown in Figure 8f, $s_{\mathbb{S}}$ is always entered in case of a violation. This reflects the assumption that the SUO could be in any state after a deviation. Theorem 7 indicates that multiple unique sequences of expected behavior can further improve a monitors confidence when identifying deviations. $\mathcal{R}_{\text{2-e-b}}$ uses 2 of such sequences.

$$\mathcal{R}_{\text{e-b}}(\mathbb{S}_{in}, e) = \begin{cases} \delta_{\mathcal{L}}(\mathbb{S}_{in}, e), & \text{if } \delta_{\mathcal{L}}(\mathbb{S}_{in}, e) \neq \emptyset \\ \mathbb{S}_{C}, & \text{otherwise} \end{cases} \quad (20)$$

A variety of resumption algorithms have been introduced. And this enumeration is not conclusive. More could be created to match specific requirements. The next section presents an evaluation framework that identifies strength and weaknesses in the algorithms' performance. It can be used to judge the algorithm for a given application scenario and identify the best.

V. Evaluation

This section presents an evaluation of the introduced method for automatic resumption of runtime verification monitors. We have already given proof that all unexpected behaviors can always be found and used to encircle the detectable deviations in Theorems 2 and 5. They are also confirmed by the collected raw data. The evaluation examines how well the presented resumption algorithms perform for different SMs and kinds of deviations. If the monitor's uncertainty of the SUO's state is reduced to a single state without missing a deviation, a detected unexpected behavior equals a deviation. Therefore, if $\text{M}_{\mathcal{E}}$ can identify deviations well, it performs a good resumption.

A. Evaluation Framework

An overview of the evaluation setup is depicted in Figure 9. The framework is employed to compare the presented resumption algorithms. A specific *application scenario* usually provides the specification and, thereby, a *reference model*. However, to make statements about the general performance of the algorithms, a generator creates the models. A single

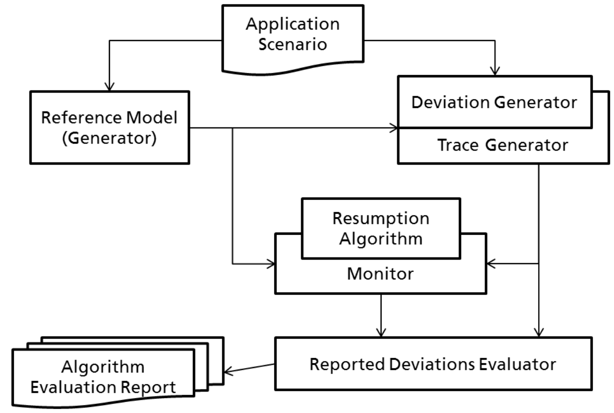


Figure 9. Overview of the evaluation framework for resumption algorithms.

state is used as initial seed. In each iteration step, one state is randomly selected and replaced by a randomly chosen number of states. The input and output transitions of the replaced state are assigned to random states in the subset. The subset of states is then connected with transitions, guaranteeing that each state is reachable from an incoming transition and can reach an outgoing transition of the original state. Otherwise, the states are connected randomly. The used set of events changes with each iteration step, i.e., some events are removed or some new events are added. This is repeated to create SMs of various sizes. The resulting SMs use global events across the whole machine and local groups. To classify the SMs, different metrics of their structure are collected, e.g., number of states and uniqueness. *Uniqueness* is the likelihood of an occurring event being unique. It is approximated by the fraction of all transitions in the SM that have a unique event.

For each reference model, multiple traces are generated by randomly selecting paths from SM′. The *deviation generator* creates SM′ by adding new states and transitions to SM for the deviations shown in Figure 10. The added transitions use undefined events ($\chi \notin \mathbb{E}^{q_s}$) of the source state q_s. The different deviations are characterized by their choice of transition targets q_t: superfluous ($q_t = q_s$), altered ($\exists e : \delta_{\mathcal{L}}(q_s, e) \mapsto q_t$), skipped ($\exists e : \delta_{\mathcal{L}}(q_s, e) \mapsto q_i \wedge \delta_{\mathcal{L}}(\chi, q_i) \mapsto q_t$) and random events ($q_t \in \mathbb{S}_{\mathcal{L}}$). The generated transitions for deviations are equivalent to faults in a real implementation. If a scenario expects more complex deviations, they can be simulated by combining these deviations. However, to evaluate the influence of each deviation kind, we apply only one kind of deviation per trace. For later analyses, the injected deviations are marked in the meta-data of the trace, which is invisible to the monitor.

The traces are eventually checked using SM extended with the individual $\mathcal{R}$. For the evaluation, our Eclipse-based tool DANA was used and extended. It is capable of using reference models as monitors [7]. Using hooks in its model execution runtime, resumption is injected if needed. Thereby, all introduced algorithms can easily be exchanged. In addition, we include $\mathcal{R}_{\text{l-c}}$, an offline *least changes* (see Section III) algorithm, for comparison.

The goal of the evaluation framework is to measure how well a monitor is at finding multiple deviations in a given application scenario. Therefore, the *reported deviations evaluator* rates each algorithm's performance by comparing the

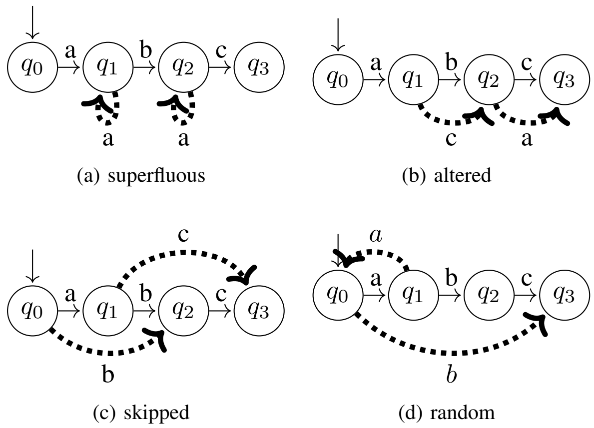


Figure 10. Examples illustrating the different deviation modules used in evaluation. The deviations are shown by dotted arrows.

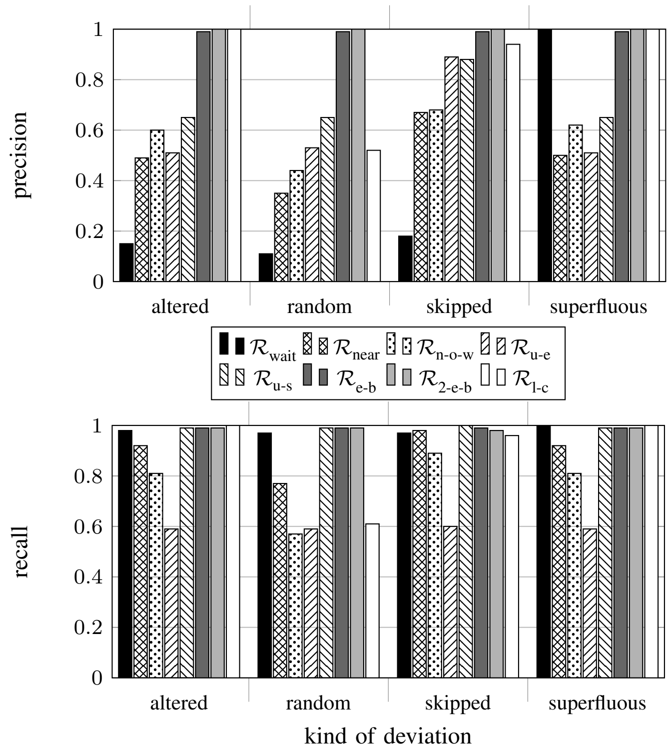


Figure 11. Precision and recall of $\mathcal{R}$ compared for different kinds of deviations.

detected and the injected deviations. It calculates for each extended monitor the well-established metrics from information retrieval: *precision* and *recall* [28][29]. Precision (21) is the fraction of reported deviations (rd) that were true (td), i.e., injected by the deviation generator. Recall (22) is the fraction of injected deviations that were reported. Both values are combined to their harmonic mean, also known as F_1 score (23).

$$p = |td \cap rd|/|rd| \tag{21}$$

$$r = |td \cap rd|/|td| \tag{22}$$

$$F_1 = 2 \cdot \frac{p \cdot r}{p + r} \tag{23}$$

A monitor that reports only and all true deviations has a perfect precision $p = 1$ and recall $r = 1$. Up to the first deviation, all extended monitors exhibit this precision, as they work like regular monitors in this case. Regular monitors only maintain this precision by ignoring everything that follows. Extended monitors may lose precision as they attempt to find further deviations. Therefore, recall estimates how likely all true deviations are reported. A regular monitor reports only the first deviation; thus, its recall is $|td|^{-1}$.

B. Comparison of Resumption Algorithms

The subscription service example (cf. Figure 2) evaluates to the F_1 scores: $\mathcal{R}_{\text{wait}} \mapsto 0.56$, $\mathcal{R}_{\text{near}} \mapsto 0.71$, $\mathcal{R}_{\text{n-o-w}} \mapsto 0.82$, $\mathcal{R}_{\text{u-e}} \mapsto 0.82$, $\mathcal{R}_{\text{u-s}} \mapsto 0.81$, $\mathcal{R}_{\text{e-b}} \mapsto 0.99$, $\mathcal{R}_{\text{2-e-b}} \mapsto 0.99$,$\mathcal{R}_{\text{l-c}} = 0.93$. For the general evaluation, traces with a total of 80 million deviations in 220 different SMs with up to 360 states have been generated and were analyzed by monitors extended with the algorithms. Each trace included 20 injected deviations on average, so the recall for a monitor reporting only the first deviation is 0.05 and its F_1 score 0.095. Figure 11 shows the precision and recall for each $\mathcal{R}$ per kind of deviation. While $\mathcal{R}_{\text{wait}}$ has the worst precision for most deviations, it shows very high recall scores overall and a perfect result for superfluous deviations. Besides that, each algorithm performs very similar for altered and superfluous deviations. When comparing $\mathcal{R}_{\text{near}}$ and $\mathcal{R}_{\text{n-o-w}}$, the former has slightly less precision; however, it provides a better recall. $\mathcal{R}_{\text{u-e}}$ has a low recall independent of the kind of deviation but also a good precision for skipped deviations. Overall, $\mathcal{R}_{\text{u-s}}$ has a high recall. The offline algorithm $\mathcal{R}_{\text{l-c}}$ has a perfect result for superfluous and altered deviations. For skipped deviations, it also provides the missing event very reliably. While hardly visible at the scale of the figure, $\mathcal{R}_{\text{e-b}}$'s precision of 0.9878 was improved to 0.9995 by a second unique sequence in $\mathcal{R}_{\text{2-e-b}}$.

Figure 12 compares the F_1 scores of the algorithms for different levels of uniqueness and numbers of states of the generated SMs. For clarity, SMs are grouped into buckets based on the metrics and the scores are averaged for each bucket. This allows a quick comparison of the algorithms, but hides the distribution of the scores across the evaluated SMs. These details are visible in the scatter plots in the Appendix, Figure 16. For example, $\mathcal{R}_{\text{2-e-b}}$ performed nearly perfect for all tested scenarios, whereas the results of $\mathcal{R}_{\text{near}}$ are much more distributed. The low overall score of $\mathcal{R}_{\text{wait}}$ is clearly visible for both metrics. For SMs with low uniqueness, $\mathcal{R}_{\text{u-s}}$ outperforms many other algorithms. However, its F_1 score slightly drops with increased uniqueness. The other algorithms benefit from an increase of uniqueness, especially $\mathcal{R}_{\text{u-e}}$. For very high uniqueness, $\mathcal{R}_{\text{u-s}}$ and $\mathcal{R}_{\text{u-e}}$ are identical. Nevertheless, both $\mathcal{R}_{\text{n-o-w}}$ and $\mathcal{R}_{\text{near}}$ perform better in this case. An increase of the state count leads to a declined performance for $\mathcal{R}_{\text{u-e}}$, $\mathcal{R}_{\text{n-o-w}}$ and $\mathcal{R}_{\text{near}}$. $\mathcal{R}_{\text{u-e}}$ even drops below $\mathcal{R}_{\text{wait}}$. $\mathcal{R}_{\text{e-b}}$ and $\mathcal{R}_{\text{2-e-b}}$ are hardly affected by state count and uniqueness and provide a near perfect overall performance. The slight advantage of $\mathcal{R}_{\text{2-e-b}}$ can be seen by the small decline of $\mathcal{R}_{\text{e-b}}$ for low uniqueness and high state counts. $\mathcal{R}_{\text{l-c}}$ cannot match this performance, but still excels the remaining algorithms. For higher state counts, its score is comparable to $\mathcal{R}_{\text{u-s}}$.

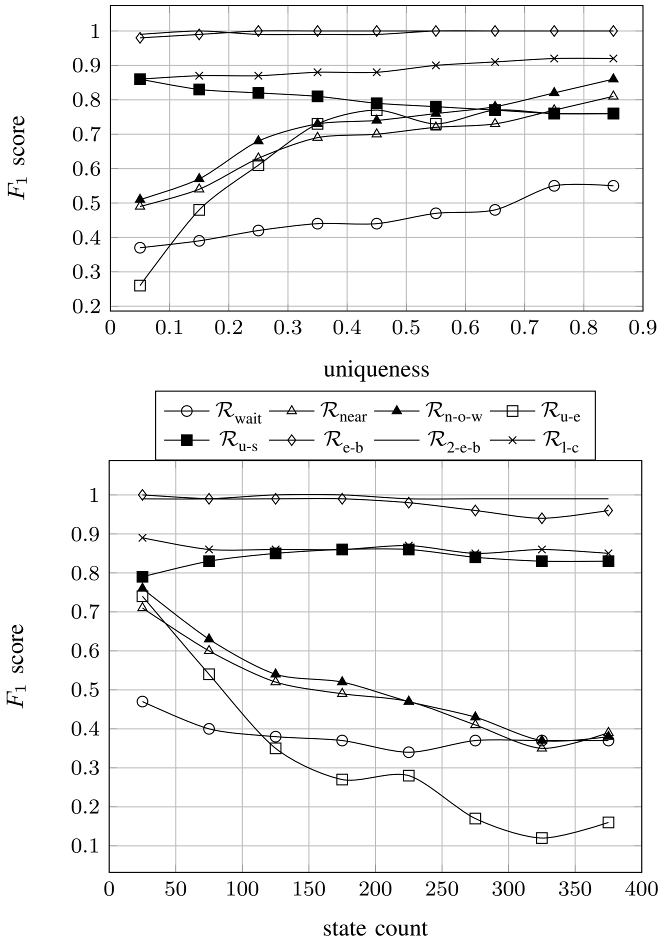


Figure 12. F_1 scores of $\mathcal{R}$ compared for metrics uniqueness and state count.

C. Discussion

In this evaluation, $\mathcal{R}_{\text{1-c}}$ is outperformed by $\mathcal{R}_{\text{e-b}}$ and $\mathcal{R}_{\text{2-e-b}}$. They perform almost perfectly for all tested scenarios. So does $\mathcal{R}_{\text{1-c}}$ for deviations matching its edits. However, it is challenged by random deviations. For example, a random deviation can transition out of a dead end or a completely different part of SM. $\mathcal{R}_{\text{1-c}}$ can only match this, if it steps back in its search space and introduces further edits. In contrast, $\mathcal{R}_{\text{e-b}}$ and $\mathcal{R}_{\text{2-e-b}}$ can match such behaviors. The slight improvement with $\mathcal{R}_{\text{2-e-b}}$ also shows that a higher precision for detecting deviations can be reached by requiring more than a single unique sequence to resume verification. While we could provide an upper bound for the worst case space and time requirements of $\mathcal{R}_{\text{e-b}}$, which is equal to segmenting the trace by unexpected behaviors, sometimes more efficient algorithms are desired.

The perfect precision and recall of $\mathcal{R}_{\text{wait}}$ for superfluous deviations were as expected, since this deviation matches exactly the resumption behavior of the algorithm. This shows that knowing the kind of deviation expected in a scenario can help formulate specialized, highly efficient algorithms. However, $\mathcal{R}_{\text{wait}}$ performs worst for all other kinds of deviations, as the SUO transitioned already internally to a different state and would have to return to its original state. It benefits from unique events, because they prevent taking wrong transitions in the meantime.

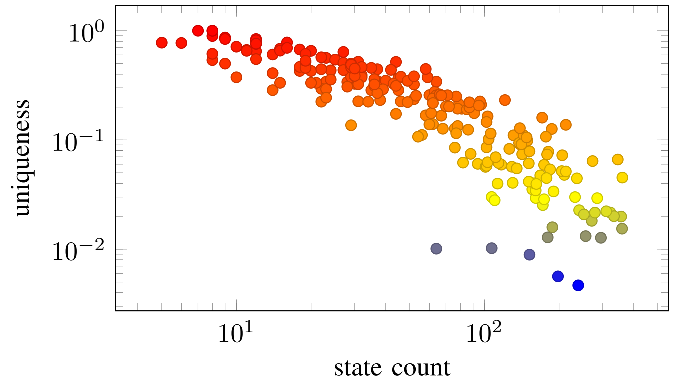


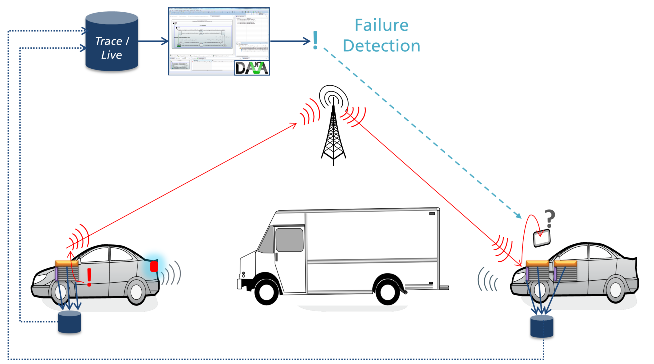
Figure 13. Scatter plot comparing uniqueness and state count of the state machines used for evaluation.

The metric uniqueness can be used as indication for the class of algorithm that is needed for a scenario. For low values, the algorithm needs to combine multiple events in order to reliably synchronize model and SUO. Therefore, in this case, algorithms should be preferred that take multiple events into account, e.g., $\mathcal{R}_{\text{u-s}}$. However, $\mathcal{R}_{\text{u-s}}$ slightly drops its precision with increasing uniqueness, as the chance increases to overeagerly synchronize with an erroneous unique event. For example, if all events are unique, any observed deviation is a unique event and the algorithm will resume with the associated state. As the next valid event is unique again, the monitor will jump back. However, in this case it registered two deviations when there actually was only one. The same holds for $\mathcal{R}_{\text{u-e}}$. Therefore, especially with a high uniqueness, it may be desirable to limit the number of options for which an algorithm may resume and use a local resumption algorithm instead. The choice between $\mathcal{R}_{\text{near}}$ and $\mathcal{R}_{\text{n-o-w}}$ depends on the desired precision and recall. According to the F_1 score, $\mathcal{R}_{\text{n-o-w}}$ is slightly favorable. However, as these algorithms may maneuver themselves into dead-ends, they are less suited for higher state counts. A bias towards lower uniqueness for higher state counts in the sample set severs the impact on $\mathcal{R}_{\text{u-e}}$. This bias is indicated by the diagonal arrangement in the comparison of uniqueness and state count for the evaluated SMs depicted in Figure 13. Nevertheless, in all cases, the F_1 scores of the extended monitors always show better results than for a regular monitor.

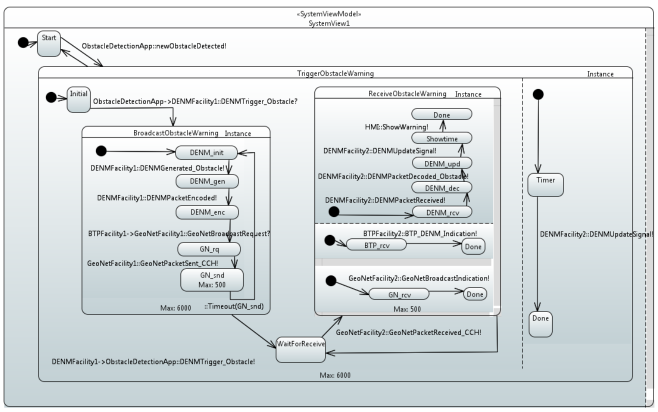
The results for the subscription service example (uniqueness 0.43, 4 states) and the respective results from Figure 12 match well. While the evaluation framework can be used to identify the best suited algorithm, this example shows that the metrics state count and uniqueness can be used as indicators for such a selection.

VI. Real World Application Scenarios

In this section, we provide examples where resumption has been applied successfully to real world use cases. DANA is the platform for description and analysis of networked applications that the presented resumption concepts were implemented in. Previous work has already shown how the platform can be used to analyze various in-vehicle infotainment functions at runtime, e.g., an *auxiliary input* service [7], and a *parking assistance* service [30]. In both cases, preliminary versions of the resumption algorithms were employed.

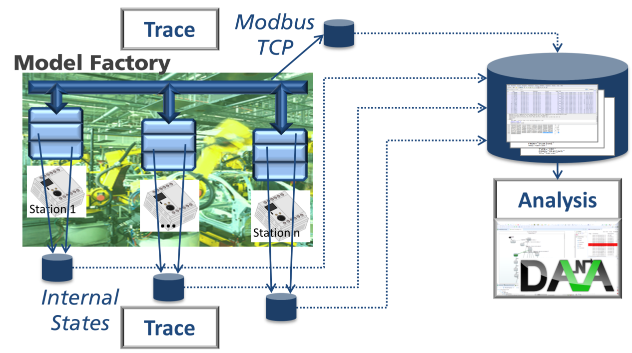


(a) Illustration of the use case.

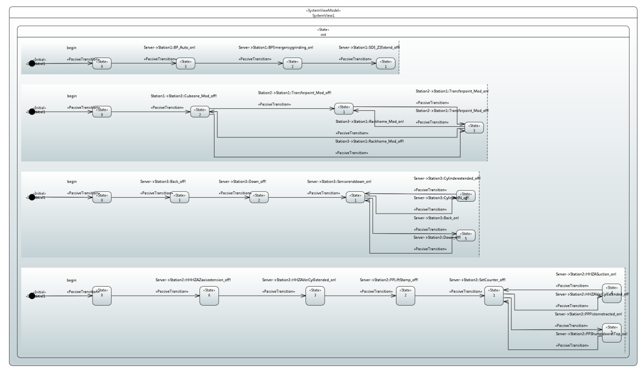


(b) Behavioral model for the use case.

Figure 14. Use case of a hazard warning application.



(a) Illustration of the use case.



(b) Behavioral model for the use case.

Figure 15. Use case of a small industrial plant.

In a more recent use case, we successfully employ resumption to support the development of a *hazard warning application*. A sudden obstacle in traffic can be dangerous; especially, if drivers realize the obstacle too late. A hazard warning can help to inform drivers in time. For example, in Figure 14a the driver in the car on the left notices an obstacle and brakes hard. As the view of the right car's driver is occluded by the van in the middle, she would only be able to notice the reaction of the van in front. With a hazard warning message from the front car, she could start braking immediately. However, such an application involves multiple cars, thus, multiple systems have to be considered. DANA is also capable to address such distributed networked systems, e.g., connected cars. We can capture the behavior to be verified of all involved cars in a model and use this for verification. Hooks in the used communication stack for car-to-car communication are employed to monitor the different communication layers. Thereby, deviations in the hazard warning implementations can be identified. For instance, the reason why a hazard warning was not displayed in the receiving car can easily be located by monitoring the progress of the animated state machine. Resumption helps to analyze the underlying problem: By providing a model containing the expected interactions, all violations of a single run can be identified.

Besides the automotive domain, resumption has been successfully applied to other application scenarios. Figure 15a shows a small industrial plant composed of three stations. The plant assembles cubes from two halves. The first station collects parts from two magazines and checks their orientation and material. The second station joins the two halves in a hydraulic press. The third station stores the assembled cubes. For each station, the changes of internal sensors and actuators controlled by the respective station are reported. This enables monitoring the plant's operation without having to alter the original control program. The communication between the stations is recorded by tapping into the switch of the Ethernet-based Modbus TCP connection. Actually, the model shown in Figure 15b contains much more details as it was automatically learned from observed behavior, i.e., each of the states contains sub-states, which are hidden in this example for clarity. Nevertheless, the sub-states are still used for verification, while this diagram provides a comprehensive overview of the plant's overall operation. Resumption helps to overcome imperfections, which such a learned model may have. While the monitor will report unexpected behavior in the case of an imperfection, resumption can often realign model and system so that verification can continue. A developer analyzing the (falsely) reported unexpected behaviors can use this feedback to improve the behavior description.

VII. Conclusion

To conclude the paper, the contributions are summarized and the findings are discussed before an outlook to future work is provided.

A. Contributions

This work examined the identification of all differences between a trace of a system under observation (SUO) and its specification at runtime through resumable monitoring. We have shown under which conditions deviations of the SUO can be detected and when they will be missed. As the main problem of detecting deviations is the current state uncertainty, the detection of unexpected behavior was examined. Unexpected behavior is independent of the SUO's actual state. By definition, the verification only needs to consider possible states of the SUO. We could show that all occurrences of unexpected behavior can be found within space $O(|\mathrm{SM}|)$ and time $O(|\mathbb{S}|)$ per step at runtime. Such unexpected behavior is always an indication for a deviation, but there may still be deviations that cannot be detected with the available information. Using these results, we have introduced a method for extending runtime monitors with resumption. Such an extension allows a specification-based monitor to find subsequent deviations. Thereby, an existing reference model of the system can be used directly without creating a secondary specification for test purposes only. Each of the introduced resumption algorithms has its strength and weaknesses. The presented framework and metrics help to find the best suited algorithm for an application scenario. Nevertheless, *expected-behavior* is the most general case of a resumption algorithm, as it has the least assumptions on possible deviations. Identifying all unexpected behaviors guarantees that all subsequences containing detectable deviations are reported.

By the result of the evaluation, $\mathcal{R}_{\text{e-b}}$ that tests for any expected behavior is the most stable and reliable of the compared algorithms. Yet, the evaluation result is not surprising, considering it is the resumption algorithm equivalent to the general candidate function δ^+ used in subsection IV-A. Therefore, using runtime verification with $\mathcal{R}_{\text{e-b}}$ resumption is equal to segmenting a trace by unexpected behaviors for any kind of deviation. Further, this fulfills our main research goal of reporting all detectable deviations using a single monitor instance.

B. Discussion

The biggest misery for resumption shown in this paper is that the current state uncertainty can only be provably reduced by an obviously conforming merging sequence and there is no guarantee that such a sequence exists. While the evaluation has demonstrated that for many cases it is still possible to identify deviations with a high likelihood using resumption, there may be cases where it is impossible to identify deviations. The problem is not limited to resumption. How is this handled by other approaches? Some consider certain events or sequences to be trusted, e.g., model-based testing can rely on the input it provides to the SUO. A fixed initial state is another example. If such events are integrated into the specification, they can be used as a kind of checkpoint to reliably resume verification. Other approaches may only attempt to detect unexpected behavior. There are runtime verification approaches that expect the specification to be split into multiple properties. They rely on detecting trigger sequences before they verify a constraint. However, their verdict always judges trigger and constraint. Further, these triggers can get quite complex and, for example, check for necessary or sufficient conditions of the property. With many properties to test, these checks may quickly become redundant. Therefore, it may be more efficient to use a single state machine, as we could show that it can detect all unexpected behaviors. Different verdicts can be associated with missing transitions and, thereby, used to retain the categorization of unexpected behaviors provided by a set of properties.

Resumption strives to resume verification after a deviation was observed. In general, the quality of the resumption depends on carefully selecting candidates for the actual state of the SUO. This entails a strong relation to the deviations that are possible or expected. Different assumptions on the system, e.g., the kind of deviations, will lead to different selections of optimal candidates. The selection of candidates is made by a *resumption algorithm*. This replaces δ^+ from (8) with a different function. δ^+ determines the current state uncertainty for the next step. Therefore, all other findings in this paper remain untouched, even if a different algorithm is chosen. However, any deviation that does not match the candidates provided by δ^+, breaks the guarantee of detecting all unexpected behaviors. Nevertheless, this allows to use specialized algorithms for specific application scenarios. Several different resumption algorithms have been evaluated and compared with regard to how well they detect deviations. $\mathcal{R}_{\text{wait}}$ is often (unintentionally) used, as it just ignores deviating events and waits in the same state. For the general case, this is not always correct and usually very unreliable. However, if deviations are only superfluous - the SUO stays in the same state when deviating - $\mathcal{R}_{\text{wait}}$ provides perfect identification of deviations. The proof is simple, as the current state uncertainty always contains exactly one state. Thereby, we can apply Theorem 1.

C. Future Work

Currently, the resumption algorithms are designed for plain state machines. This is still useful, as many advanced design concepts for state machines, like hierarchical state machines and orthogonal regions can be directly mapped to plain state machines. The event model in the layered reference model allows the state machine to be oblivious to parameter values, which would otherwise require extended state machines. If the event model can store parameters for comparison, this implies there is a state of the event model in the parameter space in addition to the state of the state machine. However, the event model is currently not updated by resumption. This is the equivalent of using $\mathcal{R}_{\text{wait}}$ for states. Therefore, future work will extend resumption to include the parameter space, i.e., the event model.

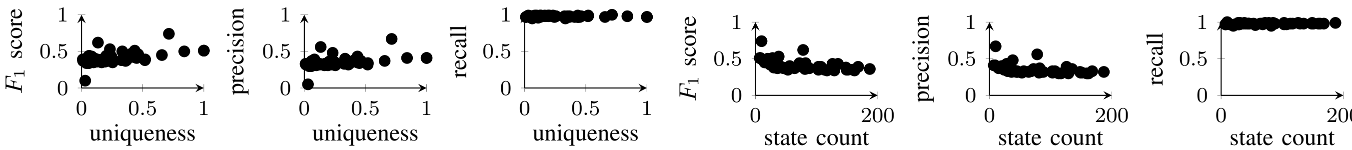
Moreover, we currently expect a total order on the observed events. In a distributed system, events can be collected in independent traces at many different sources and a global time is not always available. Therefore, obtaining a total order for all observations is not always feasible. The reference model needs to be extended to better support modeling such behavior. Possibly, a special kind of resumption can be utilized to synchronize the different traces.

Acknowledgment

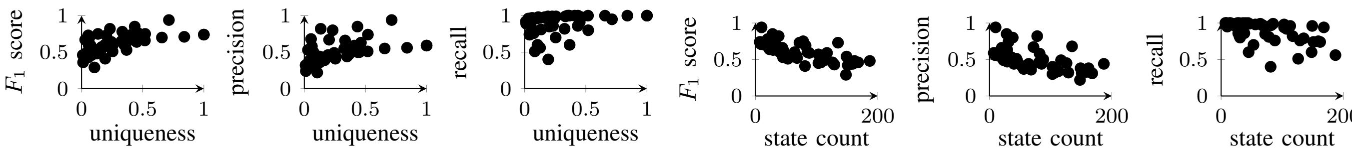
The work leading to these results has been partially funded by the Bavarian Ministry of Economic Affairs, Infrastructure, Transport and Technology.

Appendix
Detailed Results from Evaluation

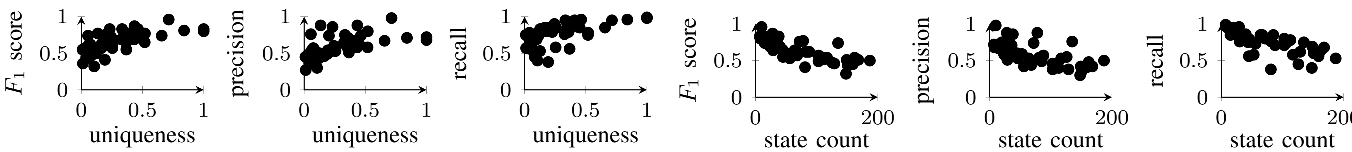
The scatter-plots in Figure 16 show the average results of the presented resumption algorithms for each machine.



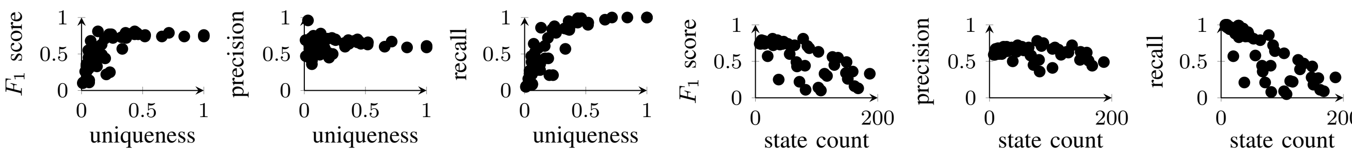
(a) $\mathcal{R}_{\text{wait}}$



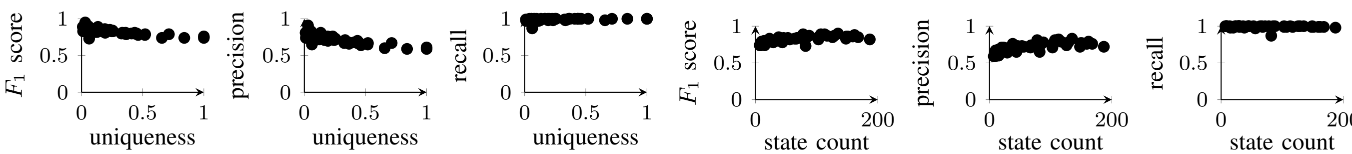
(b) $\mathcal{R}_{\text{near}}$



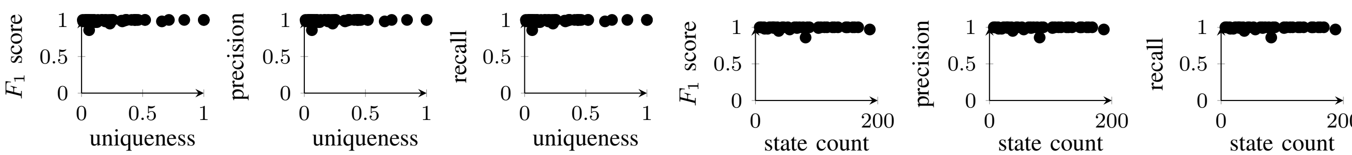
(c) $\mathcal{R}_{\text{n-o-w}}$



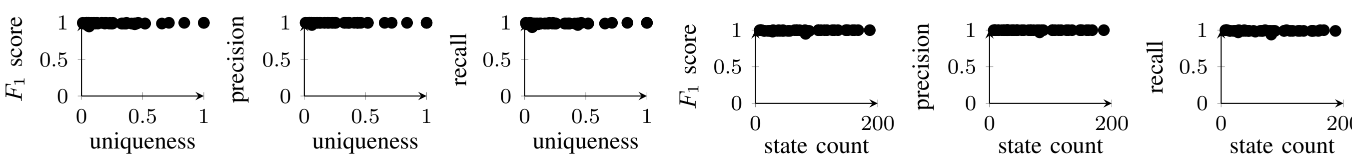
(d) $\mathcal{R}_{\text{u-e}}$



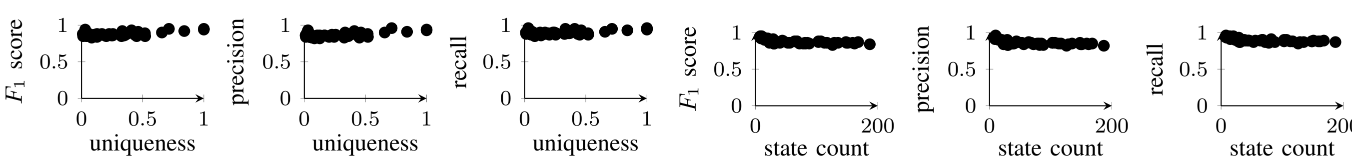
(e) $\mathcal{R}_{\text{u-s}}$



(f) $\mathcal{R}_{\text{e-b}}$



(g) $\mathcal{R}_{\text{2-e-b}}$



(h) $\mathcal{R}_{\text{l-c}}$

Figure 16. Scatter plots of the algorithms' F_1 score, precision and recall on the y axis. The x axis of the three left plots shows uniqueness, the right ones' show the state count of the evaluated machine. Each dot represents the result of all evaluations for one machine with the respective algorithm.

REFERENCES

[1] C. Drabek, G. Weiss, and B. Bauer, “Method for automatic resumption of runtime verification monitors,” in SOFTENG 2017, The Third International Conference on Advances and Trends in Software Engineering, Venice, Italy, Apr. 2017, pp. 31–36. [Online]. Available: http://www.thinkmind.org/index.php?view=article&articleid=softeng_2017_2_20_64084

[2] A. Hagemann, G. Krepinsky, and C. Wolf, “Interface construction, deployment and operation a mystery solved,” International Journal on Advances in Software, vol. 10, no. 1, 2017, pp. 61–78. [Online]. Available: https://www.thinkmind.org/index.php?view=article&articleid=soft_v10_n12_2017_5

[3] D. Heffernan, C. Macnamee, and P. Fogarty, “Runtime verification monitoring for automotive embedded systems using the ISO 26262 functional safety standard as a guide for the definition of the monitored properties,” IET Software, vol. 8, no. 5, Oct. 2014, pp. 193–203. [Online]. Available: https://doi.org/10.1049/iet-sen.2013.0236

[4] R. Rabiser, S. Guinea, M. Vierhauser, L. Baresi, and P. Grnbacher, “A comparison framework for runtime monitoring approaches,” Journal of Systems and Software, vol. 125, Mar. 2017, pp. 309–321. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0164121216302618

[5] M. Leucker and C. Schallhart, “A brief account of runtime verification,” The Journal of Logic and Algebraic Programming, vol. 78, no. 5, Mai. 2009, pp. 293–303. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1567832608000775

[6] N. Delgado, A. Gates, and S. Roach, “A taxonomy and catalog of runtime software-fault monitoring tools,” IEEE Transactions on Software Engineering, vol. 30, no. 12, Dec. 2004, pp. 859–872. [Online]. Available: https://doi.org/10.1109/TSE.2004.91

[7] C. Drabek, A. Paulic, and G. Weiss, “Reducing the verification effort for interfaces of automotive infotainment software,” SAE International, Warrendale, PA, SAE Technical Paper 2015-01-0166, Apr. 2015. [Online]. Available: http://papers.sae.org/2015-01-0166/

[8] A. Bauer, M. Leucker, and C. Schallhart, “Runtime verification for LTL and TLTL,” ACM Trans. Softw. Eng. Methodol., vol. 20, no. 4, Sep. 2011, pp. 14:1–14:64. [Online]. Available: http://doi.acm.org/10.1145/2000799.2000800

[9] Y. Falcone, K. Havelund, and G. Reger, “A tutorial on runtime verification.” Engineering Dependable Software Systems, vol. 34, 2013, pp. 141–175. [Online]. Available: http://ebooks.iospress.nl/publication/33757

[10] C. Allan et al., “Adding trace matching with free variables to AspectJ,” in Proceedings of the 20th Annual ACM SIGPLAN Conference on Object-oriented Programming, Systems, Languages, and Applications, ser. OOPSLA ’05. New York, NY, USA: ACM, 2005, pp. 345–364. [Online]. Available: http://doi.acm.org/10.1145/1094811.1094839

[11] P. O. Meredith, D. Jin, D. Griffith, F. Chen, and G. Rou, “An overview of the MOP runtime verification framework,” International Journal on Software Tools for Technology Transfer, vol. 14, no. 3, Apr. 2011, pp. 249–289. [Online]. Available: http://link.springer.com/article/10.1007/s10009-011-0198-6

[12] W. van der Aalst, A. Adriansyah, and B. van Dongen, “Replaying history on process models for conformance checking and performance analysis,” Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, vol. 2, no. 2, Mar. 2012, pp. 182–192. [Online]. Available: http://onlinelibrary.wiley.com/doi/10.1002/widm.1045/abstract

[13] J. E. Cook, C. He, and C. Ma, “Measuring behavioral correspondence to a timed concurrent model,” in IEEE International Conference on Software Maintenance, 2001. Proceedings. Florence, Italy: IEEE, 2001, pp. 332–341. [Online]. Available: https://doi.org/10.1109/ICSM.2001.972746

[14] G. Reger, “Suggesting edits to explain failing traces,” in Runtime Verification. Springer, 2015, pp. 287–293. [Online]. Available: http://link.springer.com/chapter/10.1007/978-3-319-23820-3_20

[15] C. Allauzen and M. Mohri, “3-way composition of weighted finite-state transducers,” in International Conference on Implementation and Application of Automata. Springer, 2008, pp. 262–273. [Online]. Available: http://link.springer.com/chapter/10.1007/978-3-540-70844-5_27

[16] ——, “Linear-space computation of the edit-distance between a string and a finite automaton,” arXiv:0904.4686, Apr. 2009. [Online]. Available: https://arxiv.org/abs/0904.4686

[17] J. E. Cook and A. L. Wolf, “Software process validation: Quantitatively measuring the correspondence of a process to a model,” ACM Trans. Softw. Eng. Methodol., vol. 8, no. 2, Apr. 1999, pp. 147–176. [Online]. Available: http://doi.acm.org/10.1145/304399.304401

[18] A. Pretschner and M. Leucker, “Model-based testing a glossary,” in Model-Based Testing of Reactive Systems, ser. Lecture Notes in Computer Science. Springer, Berlin, Heidelberg, 2005, pp. 607–609. [Online]. Available: https://link.springer.com/chapter/10.1007/11498490_27

[19] T. Herpel, T. Hoiss, and J. Schroeder, “Enhanced simulation-based verification and validation of automotive electronic control units,” in Electronics, Communications and Networks V, ser. Lecture Notes in Electrical Engineering, A. Hussain, Ed. Springer Singapore, 2016, no. 382, pp. 203–213. [Online]. Available: http://link.springer.com/chapter/10.1007/978-981-10-0740-8_24

[20] A. Kurtz, B. Bauer, and M. Koeberl, “Software based test automation approach using integrated signal simulation,” in SOFTENG 2016, The Second International Conference on Advances and Trends in Software Engineering, Feb. 2016, pp. 117–122. [Online]. Available: http://www.thinkmind.org/index.php?view=article&articleid=softeng_2016_5_20_65040

[21] R. L. Rivest and R. E. Schapire, “Inference of finite automata using homing sequences,” in Machine Learning: From Theory to Applications. Springer, Berlin, Heidelberg, 1993, pp. 51–73. [Online]. Available: http://link.springer.com/chapter/10.1007/3-540-56483-7_22

[22] S. Sandberg, “Homing and synchronizing sequences,” in Model-Based Testing of Reactive Systems, M. Broy, B. Jonsson, J.-P. Katoen, M. Leucker, and A. Pretschner, Eds. Springer Berlin Heidelberg, 2005, pp. 5–33. [Online]. Available: http://link.springer.com/chapter/10.1007/11498490_2

[23] S. D. Stoller et al., “Runtime verification with state estimation,” in Runtime Verification, S. Khurshid and K. Sen, Eds. Springer Berlin Heidelberg, 2012, pp. 193–207. [Online]. Available: http://link.springer.com/chapter/10.1007/978-3-642-29860-8_15

[24] M. Chupilko and A. Kamkin, “Runtime verification based on executable models: On-the-fly matching of timed traces,” EPTCS, vol. 111,, Mar. 2013, pp. 67–81. [Online]. Available: http://arxiv.org/abs/1303.1010v1

[25] T. Berg, B. Jonsson, and H. Raffelt, “Regular inference for state machines with parameters,” in Fundamental Approaches to Software Engineering, L. Baresi and R. Heckel, Eds. Springer Berlin Heidelberg, 2006, pp. 107–121. [Online]. Available: http://link.springer.com/chapter/10.1007/11693017_10

[26] A. Danese, T. Ghasempouri, and G. Pravadelli, “Automatic extraction of assertions from execution traces of behavioural models,” in Proceedings of the 2015 Design, Automation & Test in Europe Conference & Exhibition. San Jose, CA, USA: EDA Consortium, 2015, pp. 67–72. [Online]. Available: http://dl.acm.org/citation.cfm?id=2755753.2755769

[27] G. Reger, H. Barringer, and D. Rydeheard, “Automata-based pattern mining from imperfect traces,” SIGSOFT Softw. Eng. Notes, vol. 40, no. 1, Feb. 2015, pp. 1–8. [Online]. Available: http://doi.acm.org/10.1145/2693208.2693220

[28] D. M. W. Powers, “Evaluation: From precision, recall and f-measure to ROC, informedness, markedness and correlation,” Journal of Machine Learning Technologies, vol. 2, no. 1, 2011, pp. 37–63. [Online]. Available: http://dspace2.flinders.edu.au/xmlui/handle/2328/27165

[29] S. Yingchareonthawornchai, D. N. Nguyen, V. T. Valapil, S. S. Kulkarni, and M. Demirbas, “Precision, recall, and sensitivity of monitoring partially synchronous distributed systems,” in International Conference on Runtime Verification. Springer, Cham, Sep. 2016, pp. 420–435. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-319-46982-9_26

[30] C. Drabek, T. Pramsohler, M. Zeller, and G. Weiss, “Interface verification using executable reference models: An application in the automotive infotainment.” in 6th International Workshop on Model Based Architecting and Construction of Embedded Systems, ACESMB 2013. Proceedings, Miami, Florida, USA, Sep. 2013. [Online]. Available: http://ceur-ws.org/Vol-1084/paper7.pdf

Towards Virtual Reality Immersion in Software Structures: Exploring Augmented Virtuality and Speech Recognition Interfaces

Roy Oberhauser and Carsten Lecon
Computer Science Dept.
Aalen University
Aalen, Germany
email: {roy.oberhauser, carsten.lecon}@hs-aalen.de

Abstract - **Due to its abstract nature, program code structures have been inherently challenging to visualize. As virtual reality (VR) hardware products become common, their utilization for providing insights into these software structures become feasible. However, for certain user programmer-centric interaction scenarios, typical VR interfaces (controller held in each hand) can be awkward. This paper describes our VR and mixed reality (MR) fly-through software structure approach for visualizing internal program code structures and investigates additional interfaces to augment the virtuality. MR use of any keyboard and mouse are supported for programming tasks. To interface with a virtual tablet menu that is used as an oracle, a real tablet is used as a touchpad for the VR tablet. Voice control of the tablet menu was also implemented and investigated in comparison to the other interfaces. The paper evaluates the various VR and MR interfaces for their suitability for selected software development and computer science educational tasks. The evaluation results provide insight into which interfaces were more efficient and preferred by subjects.**

Keywords - Virtual reality; mixed reality; augmented virtuality; software visualization; program comprehension; software engineering; speech recognition; voice control.

I. INTRODUCTION

This paper extends [1], where we described a VR approach for visualizing, navigating, and conveying program code information interactively in a VR environment called VR-FlyThruCode (VR-FTC).

The volume of program source code produced and maintained worldwide continues to increase, yet gauging this metric is difficult since large portions are not publicly available. Google alone is estimated to have at least 2bn lines of code (LOC) internally accessible by 25K developers [2]. By some estimates well over a trillion lines of code (LOC) exist worldwide with 33bn added annually [3]. The limitations for humans to comprehend source code are evident in the relatively low code review reading rates of around 200 LOC/hour [4].

Faced with this ever-increasing code base, the question becomes: how can programmers quickly comprehend large amounts of code and understand their underlying and mostly invisible abstract structures? Common display forms used in the comprehension of source code include text, the two-dimensional Unified Modeling Language (UML), and software analysis tools. For large projects, typically multiple UML diagrams exist that are not coherently tied together and no screen can support showing all diagrams simultaneously, thus a mental patchwork of the multiple visual images is used to create a coherent model in the programmer's mind. For code files, typically these are hierarchically hidden among various subdirectories, and determining relations is an arduous task. These forms of extraction leave the programmer creating and stitching explicit or implicit visual images together.

One software pioneer, F. P. Brooks, Jr., asserted that the invisibility of software is an essential difficulty of software construction because the reality of software is not embedded in space [5]. Yet the philosopher Aristotle once stated, "thought is impossible without an image."

In 1998, Feijs and De Jong [6] presented a vision of walking through a 3D visualization of software architecture using VRML. Currently the immersive potential of VR and game engines for improving software engineering (SE) tools has still not been realized, and their practicality with off-the-shelf VR hardware remains insufficiently explored.

This paper describes our visually immersive VR approach for visualizing, navigating, and conveying program code information interactively to support exploratory, analytical, and descriptive cognitive processes [7]. In extending [1], it contributes additional interface capabilities to the VR-FTC solution concept and investigates their suitability. Specifically, the following interfaces: speech recognition for voice-directed control of the oracle – our VR voice FTC (VRVoc-FTC), as well as an MR tablet that was added to the to the MR-FTC variant to control the oracle like a touchpad. Furthermore, the fly-in theaters were replaced with an always-accessible virtual tablet which functions as an oracle – an interactive screen supplying information on request to the user. Also, as a form of augmented virtuality [8], MR support for real keyboard and mouse interfacing were added to support programming on the tablet [9], henceforth known as the MR-FTC variant. A prototype realization demonstrates the viability of these capabilities, and initial empirical experiments investigate effectiveness, efficiency, and user experience (UX) factors of the various interfaces for programming and menu navigation tasks.

The paper is organized as follows: the next section discusses related work; Section III then describes the solution approach. Section IV provides details on our prototype realization. Section V describes the evaluation of prototype and the alternative interfaces from a technical or empirical perspective. It is followed in Section VI by a conclusion.

II. Related Work

As to voice interfaces in SE, work related to voice control of SE tools includes Delimarschi et al. [10], who applied voice and gesture control to IDE tools. Lahtinen and Peltonen [11] investigated voice control for UML modeling tasks.

As to utilizing tablets in VR, Afonso et al. [12] tracked hand movement to determine if people preferred to see a virtual hand on a virtual tablet while holding a real tablet in VR. In our case, the tablet is not necessarily held (it is placed on the desk like the keyboard and mouse in a known position), we do not track hand movement, and we do not show a hand but do show a small box indicator where the tablet was last touched.

Work on visualization of software structures in VR includes Imsovision [13], which visualizes object-oriented software in VR using electromagnetic sensors attached to shutter glasses and a wand for interaction. ExplorViz [14] is a Javascript-based web application that uses WebVR to support VR exploration of 3D software cities using Oculus Rift together with Microsoft Kinect for gesture recognition.

Work regarding software visualization without the use of VR includes Teyseyre and Campo [15], who give an overview and survey of 3D software visualization tools across the various software engineering areas. Software Galaxies [16] provides a web-based visualization of dependencies among popular package managers and supports flying. Every star represents a package that is clustered by dependencies. CodeCity [17] is a 3D software visualization approach based on a city metaphor and implemented in SmallTalk on the Moose reengineering framework. Buildings represent classes, districts represent packages, and visible properties depict selected metrics, improving task correctness but slowing task completion time [18]. Rilling and Mudur [19] use a metaball metaphor (organic-like n-dimensional objects) combined with dynamic analysis of program execution. X3D-UML [20] provides 3D support with UML in planes such that classes are grouped in planes based on the package or hierarchical state machine diagrams. A case study of a 3D UML tool using Google SketchUp showed that a 3D perspective improved model comprehension and was found to be intuitive [21]. Langelier et al. [22] supports the visualization of metrics (e.g., coupling, test coverage).

In contrast to the above work, the VR-FTC approach and its variants (MR-FTC and VRVoc-FTC) leverage game engine capabilities to support an immersive VR software structure visualization environment; provide multiple dynamically-switchable (customizable) metaphors; use one VR system and controller set (without requiring gesture training) for interaction and navigation; uses a virtual tablet to provide an information screen within the VR landscape; leverages MR to support keyboard, mouse, and tablet interfaces in VR; and provides a voice direction capability to control menu options.

III. Solution Approach

As described in [1], our VR-FTC solution approach uses VR flythrough for visualizing program code structure or architecture. This inherent 3D application domain view visualization [15] arranges customizable symbols in 3D space to enable users to navigate through an alternative perspective on these often-hidden structures. For example, certain information typically not readily accessible is visualized, such as the relative size of classes (not typically visible until multiple files are opened or a UML class diagram is created), the relative size of packages to one another, and the dependencies between classes and packages.

A. Principles

The principles (P:), (basic ideas or primary methods) involved in the VR-FTC solution approach include:

P:Multiple 3D visual metaphors: Analogous to the concept of skins, it models and supports tailoring and switching between multiple code structure visualization metaphors. While our initial implementation focused on modeling and visualizing object-oriented packages, classes, and their relationships, the approach is extensible for other programming languages. Initially, two metaphors are provided "out-of-the-box" while custom mappings to other object types are supported. In the universe metaphor, each planet represents a class with its size based on the number of methods, and solar systems represent a package. Any metric can be used to map to any visual object property (like color). Multiple packages are shown by layer solar systems over one another. In the terrestrial metaphor, buildings can represent classes, building height can represent the number of methods, and glass bubbles can group classes into packages. Relationships are modeled visually as colored pipes.

P:Group metaphor: elements (classes) are grouped and delineated in a way appropriate for that language (packages for Java) and metaphor. For instance, the terrestrial metaphor uses either a glass bubble over a city or a circle of trees at the city border, and the universe metaphor uses solar systems.

P:Connection metaphor: elements (classes) are connected in a way appropriate for that metaphor. For our two metaphors, we chose colored light beams, which often are used to portray networks on a geological background.

P:Flythrough navigation: 3D navigation (motion) is provided by moving the camera in space based on controller or motion sensor input. The scenery, however, remains anchored in the scene, allowing users to remember places via their geolocation relative to other elements.

P:Oracle: a virtual tablet is provided, and can be viewed as a type of oracle to answer questions a user might have, although these cannot be formulated directly like a chatbot. Instead, it provides menus and displays source-code and SE tool-generated information as selected by the user within their given context (such as a selected object). The screens currently presented include:

- *Tags*: Setting, searching, or filtering automatic (via patterns) or manual persistent annotations/tags.
- *Source Code*: code is shown in scrollable form.

- *UML*: 2D UML diagrams can be shown on the tablet, including dynamically generated 2D diagrams, allowing users to leverage knowledge of this form if already available or if dynamic generation thereof is desired.
- *Metrics*: code metrics are displayed textually due to the large number of possible metrics that may be of interest to the user; displaying a large amount of metric information visually may be disconcerting. Customization enables metrics of interest to be utilized in a metaphor (e.g., colors, object height can relate to number of class methods, font colors can indicate a threshold is exceeded).
- *Filtering*: shows elements that match selectors.
- *Project*: change metaphors, load, or import a project.

P:MR interfaces: in addition to the VR controllers and a virtual keyboard and a virtual tablet, use of a real keyboard, mouse, and tablet used as a touch pad are supported in the MR-FTC variant.

P:Voice interface: Voice control of the tablet menu is supported in the VRVoc-FTC variant using speech recognition.

B. Process

Five steps are involved in the solution process, consisting of:

1) Modeling: modeling generic program code structures, metrics, and artifacts as well as visual objects. More details on what models and formats are used in our prototype are given in Section IV.

2) Mapping: mapping a model to a visual object metaphor.

3) Extraction: extracting a given project's structure (via source code import and parsing) and metrics.

4) Visualization: visualizing a given model instance within a metaphor.

5) Navigation: supporting navigation through the VR model instance (via camera movement based on user interaction) and navigation of the oracle menu.

IV. IMPLEMENTATION

For the implementation, we utilized the Unity engine for 3D visualization due to its multi-platform support, VR integration, and popularity, and for VR hardware both HTC Vive, a room scale VR set with a head-mounted display and two wireless handheld controllers tracked using two 'Lighthouse' base stations. First, we reiterate implementation details based on [1] and then in Sections E, F, and G we provide descriptions of the new interface implementations.

A. Architecture

Figure 1 shows the architecture. Assets are used by the Unity engine and consist of Animations, Fonts, Imported Assets (like a ComboBox), Materials (like colors and reflective textures), Media (like textures), 3D Models, Prefabs (prefabricated), Shaders (for shading of text in 3D), VR SDKs, and Scripts. Scripts consist of Basic Scripts like user interface (UI) helpers, Logic Scripts that import, parse, and load project data structures, and Controllers that react to user interaction. Logic Scripts read Configuration data about Stored Projects and the Plugin System (input in XML about how to parse source code and invocation commands). Logic Scripts can then call Tools consisting of General and Language-specific Tools. General Tools currently consist of BaseX, Graphviz, PlantUML, and Graph Layout - our own version of the KK layout algorithm [23] which we use for placing and spacing objects within a metaphor. Java-specific tools are srcML, Campwood SourceMonitor, Java Transformer (invokes Groovy scripts), and Dependency Finder. Our Plugin system enables additional tools and applications to be easily integrated.

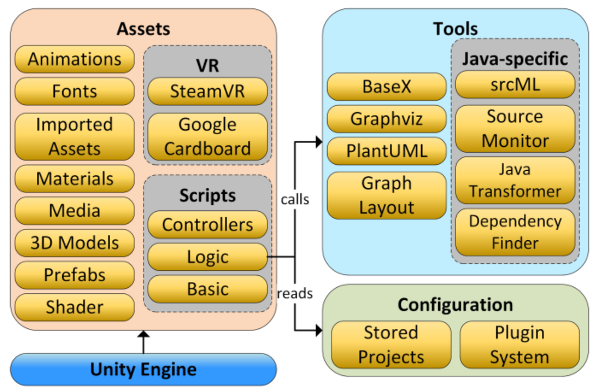


Figure 1. VR-FlyThruCode software architecture.

B. Information Extraction

For extracting existing code structure information into our model, srcML [24] is used to convert source code into XML that is then stored in the XML database BaseX, Campwood SourceMonitor, and DependencyFinder are used to extract code metrics and dependency data, and plugins with Groovy scripts and a configuration are used to integrate the various tools.

C. Project structure

For an imported project the following files are created:

- *metrics_{date}.xml*: metrics obtained from SourceMonitor and DependencyFinder are grouped by project, packages, and classes.
- *source_{date}.xml*: holds all classes in XML
- *structure_{date}.xml*: contains the project structure and dependencies utilizing the DependencyFinder.
- *swexplorer-annotations.xml*: contains user-based annotations (tags) with color, flag, and text including both manual and automatic (pattern matching) tags.
- *swexplorer-metrics-config.xml*: contains thresholds for metrics.
- *swexplorer-records.xml*: contains a record of each import of the same project done at different times with a reference to the various XML files such as source and structure for that import. This permits changing the model to different timepoints as a project evolves.

D. Metaphor Realization

To support *P:Multiple 3D visual metaphors*, a universe and a city metaphor were chosen since these are universally known and can be easily related to by users, however other metaphors are easily realizable. A welcome room similar to a cockpit, shown in Figure 2, enables the user to select the desired metaphor and project state. Figure 3 shows the city metaphor, where buildings represent classes with a label at the top and the height can portray a metric of interest such as the number of methods in that class. In the City metaphor, *P:Group* was implemented as a glass bubble over the city as shown in Figure 4. In the universe metaphor, planets represent classes and have a label in the center as shown Figure 5. *P:Group* was implemented as solar systems (see Figure 6 and compare with Figure 7). For *P:Connection*, in both metaphors colored light beams were used to show dependencies between classes or packages (see Figure 8 and Figure 9) To highlight a selected object, we utilized a 3D pointer in the form of a rotating upside-down pyramid. Graph Layout was used for placement of the visual objects, which is done automatically by the system.

To allow the user to remember objects, tagging is supported, which allows any text label to be entered and placed on an object (e.g., the 'Important' Tag in Figure 5).

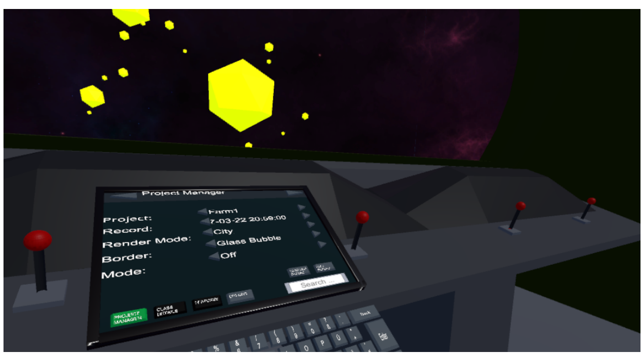


Figure 2. VR-FlyThruCode software architecture.

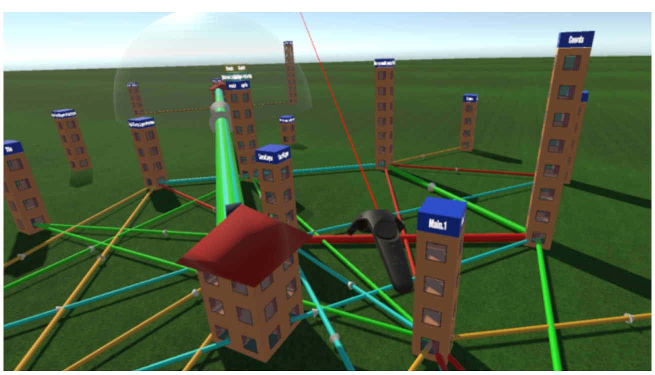


Figure 3. The city metaphor, with buildings as classes and the VR controller visible.

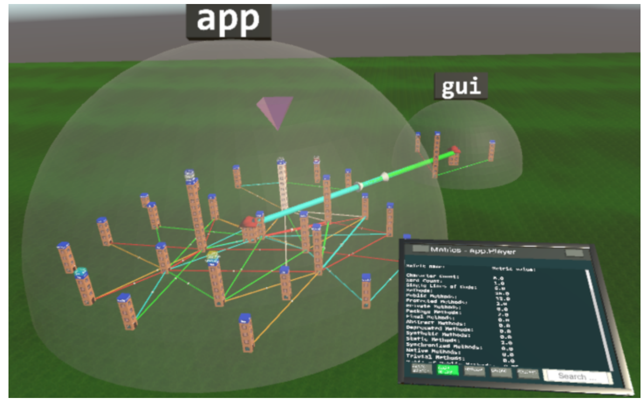


Figure 4. City metaphor showing glass bubbles with oracle visible.

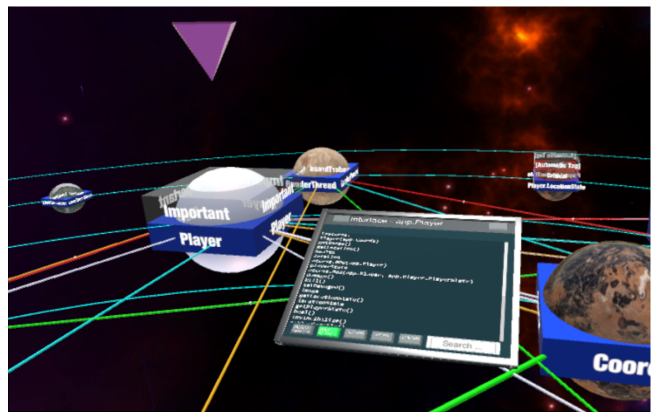


Figure 5. Universe metaphor showing tagged planet (class) and oracle.

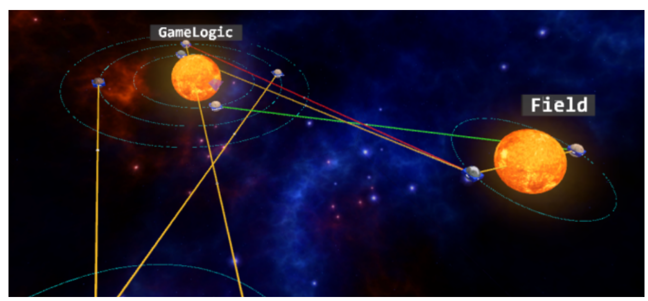


Figure 6. Universe metaphor showing solar systems.

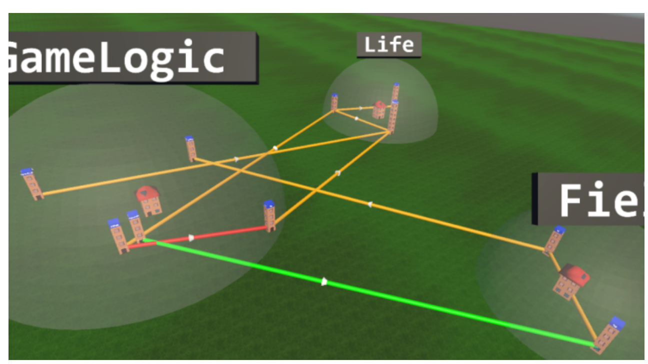


Figure 7. City metaphor showing glass bubbles.

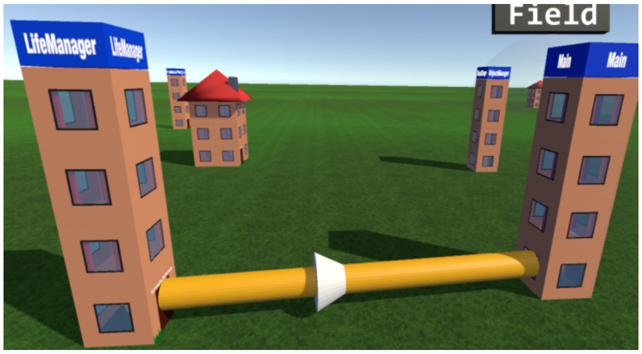


Figure 8. A directed dependency in the City metaphor.

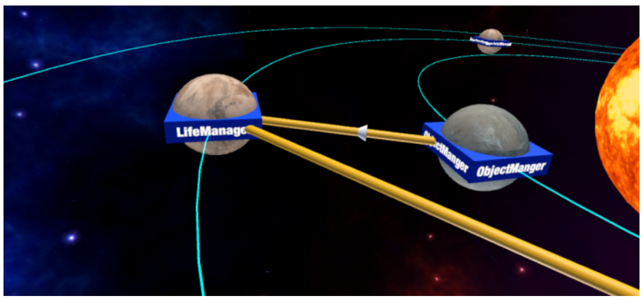


Figure 9. A directed dependency in the Universe metaphor.

To highlight a selected object, we utilized a 3D pointer in the form of a rotating upside-down pyramid (see Figure 4 and Figure 5). This was needed because, once an object is selected, after navigating to a screen or menu one may turn around and lose track of where the object was, especially if the object was small relative to its surrounding objects.

E. Virtual Reality Interaction

On the HTC Vive the touchpad on the left controller controls altitude (up, down) and the one on the right hand the direction (left, right, forward, backward), which realizes *P:Flythrough* navigation by moving the camera position. The controllers are shown in the scenery when they are within the view field, as shown in Figure 10. A virtual laser pointer was created for selecting objects, as was a virtual keyboard (see Figure 10) to support text input for searching, filtering, and tagging. To implement *P:Oracle*, menus and screens showing source code, code metrics, UML diagrams, tags, filtering, and project data are accessible via a virtual tablet.

Figure 10. VR controller using the virtual keyboard.

F. Mixed Reality Keyboard, Mouse, and Tablet

To implement *P:MR interfaces* in the MR-FTC variant, access to a real keyboard and mouse in MR was achieved via a live camera view, which was integrated into the VR landscape using a virtual plane object (see Figure 11). A Logitech C920 webcam with a 1080p resolution was used instead of the Vive Front camera to achieve better resolution. With this option, the user can utilize their favorite keyboard and mouse that they are already accustomed to.

Figure 11. MR-FTC variant with a MR keyboard (mouse out of view).

We chose to automatically activate and show MR when the user's tilts the goggles low enough, as one would if one were to wish to see the keyboard when using it and turn MR off again if one tilts the head up far enough again. Keyboard and mouse inputs are accessible at any time, not just when MR is activated.

Figure 12. Android tablet use as augmented virtuality.

To support a tablet, we created an Android app with no visible user interface (it only needs to detect the finger location as shown in Figure 12) and tested it on a Sony XPERIA Z2 Tablet with Android 6.0.1. When a user touches the App screen, a UDP packet consisting of the finger location coordinates and a tap event flag to our MR-FTC Unity application on the PC via the wireless network. A mouse pointer in the form of a white cube is then shown on the oracle (virtual tablet) at the equivalent position. A double-tap results in an OnClick-Event. Figure 12 shows a user holding the tablet and “seeing” the location on the virtual tablet. Figure 13 shows a closeup of the oracle and shows the cube indicator of the finger location. Note that no orientation information is sent (e.g., tilt angle) and that for the experiment the tablet was fixed to the desktop in a known location for the user, so they do not need to hold it. The app is blank except for displaying the coordinates because the user in VR does not see the real tablet; it is not under the camera as the MR keyboard and mouse but rather in a fixed position on the desk known to the user before putting the VR goggles on. A Sony XPERIA XZ with Android 8.0.0 was also tested.

Figure 13. Closeup of the oracle showing menus and the source code view.

G. *Voice Directed Control*

To implement a *P:Voice interface* in our VRVoc-FTC variant - specifically voice-directed control of the oracle menu, we evaluated various openly available speech recognition options. Our constraints were that we not be required to pay for any service, which constrained certain well-known cloud options. We then evaluated various popular libraries but our requirements were that little to no training would be required. After various attempts that did not result in satisfying solutions, we settled on Windows System Speech (WSS) and Unity Speech (Cortana). For the evaluation, only WSS was used since network access to Cortana was blocked by campus IT. Off-campus, Windows 10 Pro (with Fall Creators Update) was tested with Cortana.

Because WSS support is not integrated in Unity 5.6, we used a client-server program to send the commands from WSS to Unity, where it is processed by the SpeechHandler class. Thus, any Speech API (Application Programming Interface) could send the commands over TCP/IP to VR-FTC.

The microphone was mounted on a headset, so it is close to the mouth and thus reduces unrelated noise inputs. Currently no indication is given if a command is not understood, but because the response is normal fast, after a second the user should notice that the command was not executed and will likely try again.

The following words are supported at this time:

"class information", "class details": opens the view "Class Details"

"source code", "source": shows the source code for the selected class

"project manager", "project": opens the view "Project Manager"

"feature screen", "feature": opens the view "Feature Screen"

"option screen", "options screen", "option", "options": opens the configuration options view

"left", "previous": changes the view to the previous window

"right", "next": changes the view to the next window

Under Unity Cortana, we also support "search <classname>", which shows the program code of that class. In WSS this functionality was not possible since only predefined words can be used, and its free speech recognition mode performance was unusable for SE-specific tasks.

WSS does not improve automatically over time. While one can manually improve WSS, we noticed no improvements after a half hour session. Unity Speech (Cortana) probably improves automatically but we were unaware of an option to manually improve it via sessions.

V. Evaluation

After showing the feasibility of the solution with our implementation, our technical evaluation focused on assessing the implementation's viability on current VR hardware options. To compare our VR-FTC solution's suitability, effectiveness, and efficiency with non-VR and provide an overall picture, empirical evaluations were performed as indicated below. Section D includes new evaluations focused on the interfaces for programming-centric tasks, where we performed an empirical evaluation of MR-FTC with a keyboard and mouse input compared to non-VR and a virtual keyboard. And to evaluate menu interfaces of the oracle (virtual tablet), we performed an empirical study comparing VR controllers, a MR touchpad-like tablet interface, and voice control.

A. *Technical Evaluation*

Our technical evaluation performed in [1] utilized an HTC Vive with a 2160×1200 447 PPI resolution, Unity 5.3.5f1 PE, SteamVR 1479163853. The desktop PC had a 4GHz i7-6700K, 32GB RAM, SSD, NVIDIA GeForce GTX980Ti with 6GB GDDR5, Win7 Pro x64 SP1. The notebook was a MSI GS60 2.5GHz i7-4710HQ, 16GB RAM, NVIDIA GeForce GTX870M with 3GB GDDR5, SSD, Win10 Home x64, which did not meet Vive's minimum requirements but allows us to determine if a notebook (popular among software developers) would suffice for our VR application.

1) Resource usage: RAM was allocated for a 64-bit implementation was 220MB (with no project), 250MB (project with 27 classes), and 620MB (project with 95 classes). On the notebook, graphics card load was 80% without a project and went to 90% with a loaded project (for the PC 20%). We determined the CPU was the bottleneck, with load on the PC for a large project almost always at 100%. We believe that scripts attached to each visible class invoke their update method for each frame, and plan to optimize this in future work.

1) Frame rate: to determine the performance impact of each metaphor and if a notebook would be sufficient, the Saxon XSLT 2.0 and XQuery processor consisting of 300K lines of code and 1635 classes was loaded and the frames per second (FPS) measured via a custom script.

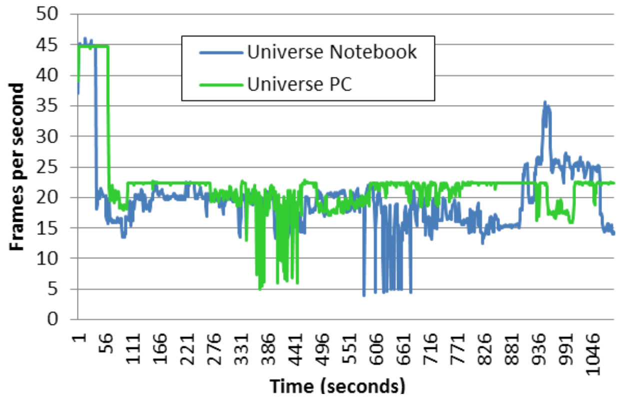


Figure 14. Universe metaphor frame rate over time on notebook and PC.

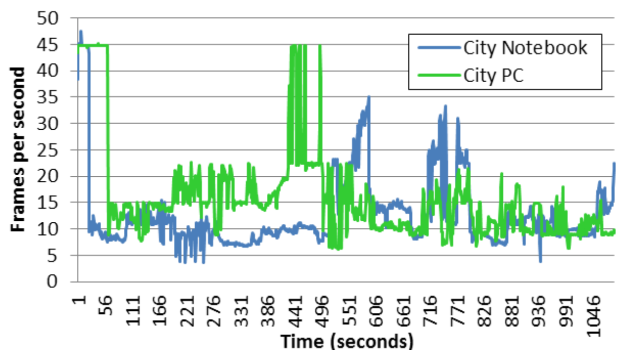


Figure 15. City metaphor frame rate over time on notebook and PC.

Figure 14 and Figure 15 show that the notebook mostly exhibited lower FPS rates than the PC, and that the city metaphor lowered the FPS rate. This is also shown by the average FPS rates for universe (notebook=20.0; PC=22.2) and city (notebook=12.7; PC=16.0). Below 15 FPS is not tolerable (early silent films had 16-20). An initial analysis found the dynamic UML generator - run in a separate process - as a main cause, and this will be addressed in future work. The universe ran better than city, since city included multiple shadows, reflections from the glass bubble, and a terrain. Higher FPS occurred when flying to an outer package such that far fewer objects were in view.

B. *Suitability of VR for SE Tasks*

Our empirical evaluation to compare the SE task suitability of VR vs. non-VR was performed in [1] using the HTC Vive. Our hypotheses were (1) that VR mode is on par with non-VR in effectiveness and efficiency for SE code structure analysis tasks and education, and (2) VR mode offers an immersive and UX quality absent in non-VR.

Resource-constraints such as having only one Vive and the time-intensive 2-on-1 supervision of the experiment with a single subject at a time limited our sample size. A convenience sample of 10 computer science students of various academic semesters (1; 3-4; 6-9 grouped respectively as beginner, intermediate, and advanced) participated and self-rated their programming and UML competency (Figure 16). Object-orientation (OO) is taught in the second and UML in the fourth semester. The one first semester student had work experience in the software industry and thus knew OO and UML. Each received a short tutorial on non-VR FTC (three had prior experience). Project A consisted of 2 packages, 27 classes, and 170 methods, while Project B had 5 packages, 95 classes, and 800 methods.

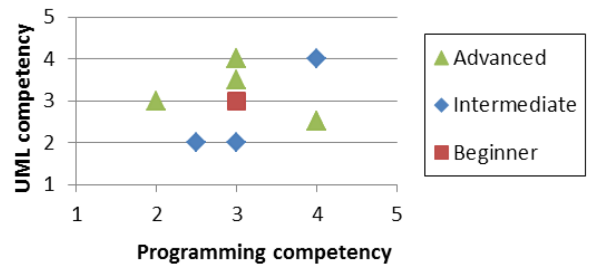


Figure 16. Participant UML/programming self-rating by semester level.

In non-VR mode, project A was loaded in the universe and thereafter the city metaphor, and likewise with B, and the same sequence repeated for VR mode. 8 questions were asked per case dealing with program code structural comprehension requiring navigation (not the same set each time), resulting in 64 questions (see Figure 17); 5 additional general questions followed giving 69 in total. So that the VR glasses need not be removed, and in order not to skew the task durations in non-VR mode, questions were asked and answered verbally and noted by a supervisor.

1) How many connections/dependencies does class X have within the package Foo?
2) How many connections/dependencies does class Y have within the package Bar?
3) Add a tag to the class X
4) Which package is the largest/smallest?
5) How many connections/dependencies does package Foo have?
6) How many connections/dependencies does package Bar have?
7) How many variables are declared in the class Y in package Foo?
8) Which classes are directly connected with the class Y?
9) Name all classes on the shortest path from A to B.
10) How many overloaded functions does the class Z have in package Bar?
11) In what package did you set your tag?

Figure 17. Sample timed task questions and requests.

As to efficiency, on average 92.5 min were needed for the 64 questions, 43.4 in VR mode vs. 39.5 min in non-VR (10% difference), while VR training took 9.4 min. Figure 18 shows the sum of the task durations for each mode per subject, whereby subjects 8-10 had prior FTC familiarity. Although VR mode was 10% slower, this was their first experience using VR. In addition, in non-VR mode the HUD is instantly available and screens can be switched, while in VR mode navigation to a screen is required. In our opinion, more VR practice might reduce this difference further.

With regard to effectiveness, given 32 questions in each mode across 10 subjects, in non-VR 300 (94%) and in VR 296 (93%) were answered correctly.

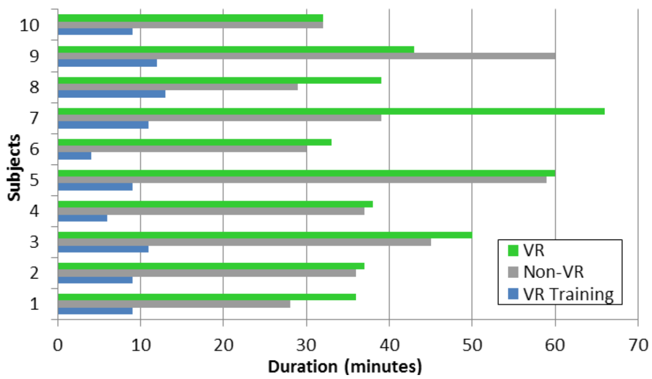


Figure 18. Sum of task durations per subject for VR and non-VR modes.

Subjects considered both FTC application modes suitable for these SE tasks. Comments included liking how information was visually displayed, its closeness to a reality, its clear arrangement, and that head movement could be used for exploring (which non-VR cannot provide). Subjects felt no differently after using non-VR, whereas after VR the feeling was described as impressive for seven of the ten subjects. The other three subjects reported VR sickness symptoms, a type of visually-induced motion sickness exhibiting disorientation. We plan to address the VR sickness in future work, e.g., by increasing the frame rate via optimizations and reducing the speed of camera movement.

C. *Mixed-Reality Interface for SE Tasks*

An empirical evaluation of the MR-FTC keyboard capability using a convenience sample consisting of Computer Science students was performed as described in [9]. For evaluating typing speed in particular, for program text such as comments which are full words without special characters, five subjects were required to write two unique pangrams consisting of 18 words using a text editor (Notepad++), the MR keyboard, and the VR only keyboard. We varied the starting configuration order among the subjects to minimize training effects. As shown in Table I, the text editor was the most efficient with 50 seconds duration and 22.5 words per minute (wpm) with an average error rate of 3.3%. With MR 75 seconds were required (16.0 wpm) with an error rate of 3.3%. With the VR keyboard 110 seconds were required (10.1 wpm) with an error rate of 4.4%. Thus, the MR keyboard was faster than the VR keyboard and did not exhibit a higher error rate. However, the subjects needed 11 seconds on average between laying down the VR controllers and pressing the first letter on the keyboard.

TABLE I. TEXT EDITOR, MR, AND VR PANGRAM MEASUREMENTS (AVERAGE)

	Text Editor	MR	VR
Duration (seconds)	50	75	110
Words per minute	22.5	16.0	10.1
Error rate	3.3%	3.3%	4.4%

For evaluating programming, four subjects were required to view a certain class and then create a class and were given certain specified modifications thereafter (creating some object and setting some variable to some value) using either a text editor or the MR keyboard. As seen in Table II, using a text editor (Notepad++), they needed on average 50 seconds to analyze a similar class, 30 seconds to create a new class, and 144 seconds to do the programming. Using the MR keyboard, they needed 84 seconds to analyze a similar class, 77 seconds to create a class, and 245 seconds to complete the programming.

TABLE II. TEXT EDITOR AND MR MEASUREMENTS (AVERAGE IN SECONDS)

	Analysis	Class Creation	Programming
Text editor	50	30	144
MR	84	77	245

We were pleased that none of the subjects reported motion sickness despite the inclusion of MR and the average response to how they felt afterwards was 4.75 (on a scale of 1 to 5 with 5 best).

Although the keyboard was a German layout keyboard, we noted that some subjects already had used that specific keyboard model before (Logitech K280e) while others had not and thus needed more time to search for certain specific keys. In searching they needed to get close with the VR goggles to see the key label, so we will consider providing a zoom or magnification option in the interface in the future.

D. *Voice, Tablet, and Controller Interface Comparison*

To compare menu-centric control of the oracle with the VR-FTC, MR-FTC tablet, and VRVoc-FTC variants empirically, we used a convenience sample of six Computer Science students. During the experiment, one of the subjects exhibited VR sickness symptoms and could not continue, so the results for this student were removed. In future work, we will attempt various optimizations and see if this reduces the likelihood. After the supervised treatments, the subjects filled out a questionnaire and were debriefed.

To ascertain efficiency effects of the different interfaces, each subject was given five different SE tasks by a supervisor to perform in the VR-Based VR-FTC City metaphor with each interface, such as find a certain class, determine how many methods a certain class has, tag a class, add a comment. Similar tasks were given for each case of interface when using primarily Voice control (V), Tablet control (T), or VR Controller (C). A random order of treatments was applied in order to ascertain if the treatment ordering created a training effect (e.g., always faster with the second or third interface), and the order is depicted as labels in Figure 19 which shows the total task duration by subject for each interface. As can be seen, the ordering did not show a clear trend. On average, voice took 353 seconds, the tablet 346 seconds, and the VR controller 197 seconds. we approximate that the tablet and voice are similar in efficiency and averaged together (350s) they are 77% slower than controller use.

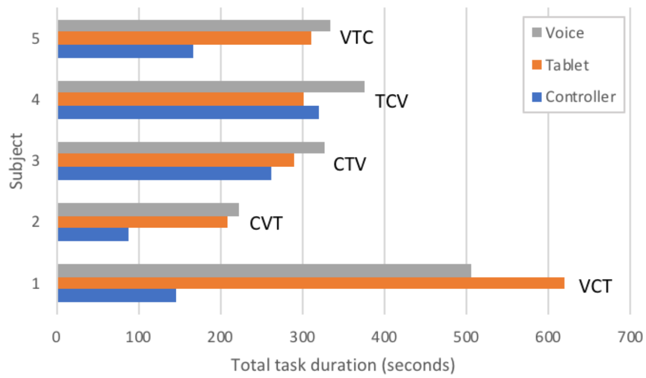


Figure 19. VR-FTC total task duration in seconds by subject for voice, tablet, and controller interfaces.

Figure 20 shows the results of the subjective assessment of suitability and enjoyment by the subjects. Overall one observes that the VR controllers had the most positive suitability and enjoyment ratings, and that voice was had three positive assessments for suitability and enjoyment. The tablet had the least positive suitability and enjoyment assessment.

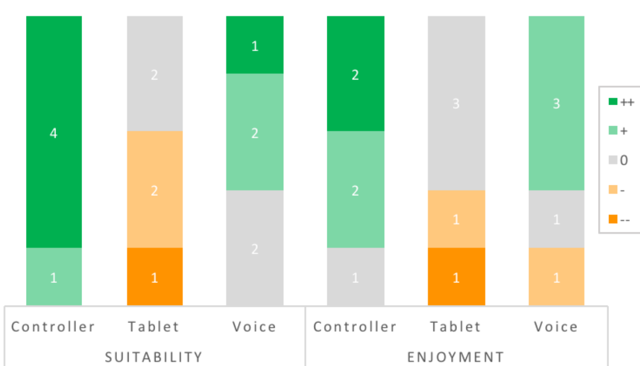


Figure 20. Suitability and enjoyment assessments in VR-FTC for voice, tablet, and controller interfaces from very high (++) to very low (--).

The supervisor gave subjects their tasks via speech, and subjects also spoke with supervisor during their tasks, which caused unintended commands to be executed. Also, the university lab setting included various other students and background noise, which reflect a realistic setting for software developers. From approximately 50 voice commands that were required to perform the SE tasks, the supervisor noted that less than 1 was not heard and less than 1 misinterpreted. For instance, if the subject was naming the functions the speech interpreter might interpret a command based on the name of a function or class. To address this, in future work we plan to provide a clear delineation for voice command mode (e.g., push to talk on a controller) and to inject the experiment directions into the scenery.

We were surprised that the MR-FTC tablet was not found to be that suitable or enjoyable for controlling the virtual tablet. We thought it would be found to be similar to a touch pad on a notebook, some users thought it was a good idea and more stable than holding the controllers.

Voice was the slowest overall. Voice direction almost always requires more time than direct control (e.g., keyboard or mouse on a PC vs. voice control), however it can free up other interfaces, and this was recognized as a benefit in the debriefing by a number of subjects.

E. Discussion

The technical evaluation of Section V.A showed suitable resource usage but pointed out frame rate issues. As to the suitability of using VR-FTC for SE tasks such as answering structural issues like those in Figure 17, Section V.B showed that while VR-FTC was 10% slower on average for untrained VR users, no significant difference in correctness were observed. Thus, our empirical hypotheses were confirmed by our results and the feedback from participants.

One threat to validity is the order effect of application usage in that non-VR followed VR. Thus, non-VR times include the overhead for gaining familiarity with the application concepts, and VR mode did not have this overhead. However, 2D monitor and mouse-centric interaction was a pre-existing competency, while VR display and navigation was a new interaction paradigm for all subjects. Furthermore, subjects 8, 9, and 10 had prior familiarity with the non-VR FTC via a prior experiment, yet their task duration times did not exhibit any clear trend that prior familiarity sped up the non-VR task durations. Furthermore, the 1% difference in correctness might be attributed to mental fatigue since VR was done in the second hour. A further threat to validity is that the positive experience is possibly a novelty effect - VR veterans would be needed to be included to assess this factor. For better external validity, the sample size should be larger and more diverse to include professionals. However, the results can be viewed as indicative and the approach as promising if we can address the VR sickness. We made optimizations for the frame rate issues to address VR sickness in further empirical studies, and only one person in those experiments experienced VR sickness.

With regard to the results in Section V.C of using FTC with a keyboard, a non-VR text editor remains more efficient, yet usage of the MR-FTC keyboard was faster than a purely VR-FTC keyboard and, once familiar with a certain keyboard, we expect the overhead of MR to be reduced to an acceptable level given sufficient practice. The overhead of switching between VR controllers to keyboard and back again can be seen as analogous to the overhead of keyboard use on a PC and moving the hand to the mouse and back again and may thus be considered acceptable for certain users. We will investigate this further in future work.

The Section V.D results regarding interfaces for oracle menu control found that the use of the VR controllers was most efficient, suitable, and enjoyable. Since VR controllers are specifically intended for interacting in VR, this result is not surprising. However, our investigation showed that the impacts of alternative interfaces may still be acceptable for certain users, and this efficiency impact is not on the order of magnitudes in scale. VRVoc-FTC indicated that it has potential, with no subject indicating it was not suitable. MR-FTC with a real tablet was not found to be suitable or enjoyable and was about as slow as VRVoc-FTC, which we found surprising since the virtual tablet is shown in VR and one would think it would be enjoyable to hold one while in VR. In future work we will investigate potential improvements to its interface and removing all need for having VR controllers when using the tablet. One threat to validity is the small sample sizes, yet it does provide some indicator as to which interfaces to pursue and investigate further and consider for industrial usage studies.

VI. Conclusion

As VR devices become more commonplace, the potential of VR to assist programmers in program comprehension can provide an immersive alternative to commonly available tools and paradigms. This paper described our VR flythrough software structure visualization approach called VR-FTC. As augmented virtuality we explored alternative interfaces to the VR controllers including MR-FTC (keyboard and tablet) and VRVoc-FTC (voice) variants. It immerses users into multiple and customizable VR metaphors for visualizing, navigating, conveying, and changing program code information interactively to support exploratory, analytical, and descriptive cognitive processes.

Our investigation observed that when comparing SE tasks in VR to non-VR, non-VR (with which the subjects are quite familiar) was more efficient. However, given more VR experience and training these differences could become smaller, and the VR efficiency overhead may be justified by the better and more enjoyable and motivational experience for users. In exploring alternative interfaces in VR, for text input we found that MR-FTC using keyboard and mouse was a viable option and faster than a virtual keyboard. For menu navigation, we found that VR controllers were most efficient and that voice, although less efficient, was an acceptable alternative option. A real tablet interface equivalent to a touchpad was not found to be suitable or enjoyable and was equivalent to voice in efficiency. However, in future work we intend to turn the tablet into a complete replacement for the VR controllers and reevaluate its suitability.

Future work includes further analysis and optimizations to address any remaining VR sickness symptoms, and comprehensive empirical studies in the industry.

Acknowledgment

The authors would like to thank Dominik Bergen, Sinan Emecan, Alexandre Matic, Lisa Philipp, and Camil Pogolski, and Patrick Sprenger for their assistance with various aspects of the design, implementation, and evaluation.

References

[1] R. Oberhauser and C. Lecon, "Immersed in Software Structures: A Virtual Reality Approach," Proc. of the Tenth International Conference on Advances in Computer-Human Interactions (ACHI 2017). IARIA, 2017, pp. 181-186.

[2] C. Metz, *Google Is 2 Billion Lines of Code—And It's All in One Place*. [retrieved: May, 2018]. Available from: http://www.wired.com/2015/09/google-2-billion-lines-codeand-one-place/

[3] G. Booch, "The complexity of programming models," Keynote talk at AOSD 2005, Chicago, IL, Mar. 14-18, 2005.

[4] C. F. Kemerer and M. C. Paulk, "The impact of design and code reviews on software quality: An empirical study based on PSP data," IEEE Trans. on Software Engineering, vol. 35, no. 4, 2009, pp. 534-550.

[5] F. P. Brooks, Jr., The Mythical Man-Month. Boston, MA: Addison-Wesley Longman Publ. Co., Inc., 1995.

[6] L. Feijs and R. De Jong, "3D visualization of software architectures," Comm. of the ACM, 41(12), 1998, pp. 73-78.

[7] D. M. Butler et al., "Visualization reference models," in Proc. Visualization '93 Conf., IEEE CS Press, 1993, pp. 337–342.

[8] P. Milgram, H. Takemura, A. Utsumi, and F. Kishino, "Augmented reality: A class of displays on the reality-virtuality continuum," In: Telemanipulator and telepresence technologies, Vol. 2351. International Society for Optics and Photonics, 1995, pp. 282-293.

[9] R. Oberhauser, "Immersive Coding: A Virtual and Mixed Reality Environment for Programmers," In: Proceedings of The Twelfth International Conference on Software Engineering Advances (ICSEA 2017), IARIA XPS Press, 2017, pp. 250-255.

[10] D. Delimarschi, G. Swartzendruber, and H. Kagdi, "Enabling integrated development environments with natural user interface interactions," Proceedings of the 22nd International Conference on Program Comprehension (ICPC 2014). ACM, 2014, pp. 126-129.

[11] S. Lahtinen and J. Peltonen, "Enhancing usability of UML CASE-tools with speech recognition," Proceedings of the 2003 IEEE Symposium on Human Centric Computing Languages and Environments. IEEE, 2003, pp. 227-235.

[12] L. Afonso, P. Dias, C. Ferreira and B. S. Santos, "Effect of hand-avatar in a selection task using a tablet as input device in an immersive virtual environment," 2017 IEEE Symposium on 3D User Interfaces (3DUI). IEEE, 2017, pp. 247-248.

[13] J. I. Maletic, J. Leigh, and A. Marcus, "Visualizing software in an immersive virtual reality environment," 23rd Intl. Conf. on Softw. Eng. (ICSE 2001) Vol. 1., IEEE, 2001, pp. 12-13.

[14] F. Fittkau, A. Krause, and W. Hasselbring, "Exploring software cities in virtual reality," IEEE 3rd Working Conference on Software Visualization (VISSOFT), IEEE, 2015, pp. 130-134.

[15] A. R. Teyseyre and M. R. Campo, "An overview of 3D software visualization," Visualization and Computer Graphics, IEEE Trans. on, vol. 15, no. 1, 2009, pp. 87-105.

[16] A. Kashcha. *Software Galaxies* [retrieved: May, 2018]. Available: http://github.com/anvaka/pm/

[17] R. Wettel and M. Lanza, "Program comprehension through software habitability," in Proc. 15th IEEE Int'l Conf. on Program Comprehension, IEEE CS, 2007, pp. 231–240.

[18] R. Wettel et al., "Software systems as cities: A controlled experiment," in Proc. of the 33rd Int'l Conf. on Software Engineering, ACM, 2011, pp. 551-560.

[19] J. Rilling and S. P. Mudur, "On the use of metaballs to visually map source code structures and analysis results onto 3d space," in Proc. 9th Work. Conf. on Reverse Engineering, IEEE, 2002, pp. 299-308.

[20] P. M. McIntosh, "X3D-UML: user-centred design, implementation and evaluation of 3D UML using X3D," Ph.D. dissertation, RMIT University, 2009.

[21] A. Krolovitsch and L. Nilsson, "3D Visualization for Model Comprehension: A Case Study Conducted at Ericsson AB," University of Gothenburg, Sweden, 2009.

[22] G. Langelier et al., "Visualization-based analysis of quality for large-scale software systems," in Proc. 20th Int. Conf. on Automated Software Engineering, ACM, 2005, pp. 214-223.

[23] T. Kamada and S. Kawai, "An algorithm for drawing general undirected graphs," Information processing letters, 31(1), 1989, pp. 7-15.

[24] J. Maletic et al, "Source code files as structured documents," in Proc. 10th Int. Workshop on Program Comprehension, IEEE, 2002, pp. 289-292.

A Study of Cordova and Its Data Storage Strategies

Gilles Callebaut*, Michiel Willocx†, Jan Vossaert†, Vincent Naessens†, Lieven De Strycker*

*KU Leuven
DRAMCO, Department of Electrical Engineering (ESAT),
Ghent Technology Campus, 9000 Ghent, Belgium
{gilles.callebaut, lieven.destrycker}@kuleuven.be
†KU Leuven
MSEC, imec-DistriNet
Ghent Technology Campus, 9000 Ghent, Belgium
{michiel.willocx, jan.vossaert, vincent.naessens}@cs.kuleuven.be

***Abstract*—The mobile world is fragmented by a variety of mobile platforms, e.g., Android, iOS and Windows Phone. While native applications can fully exploit the features of a particular mobile platform, limited or no code can be shared between the different implementations. Cross-Platform Tools (CPTs) allow developers to target multiple platforms using a single codebase. These tools provide general interfaces on top of the native Application Programming Interfaces (APIs). Apart from the performance impact, this additional layer may also result in the suboptimal use of native APIs. More specifically, this paper focuses on Apache Cordova; the most used CPT. Via a data storage case study, the impact of the abstraction layer is analyzed. Both the performance overhead and API coverage are discussed. Based on the analysis, an extension to the cross-platform storage API is proposed and implemented. In addition, the Cordova framework, including the employed bridge techniques, is studied and elaborated.**

***Keywords*–Cross-Platform Tools; data storage; performance analysis; API coverage; Apache Cordova/Phonegap.**

I. Introduction

An increasing number of service providers are making their services available via the smartphone. Mobile applications are used to attract new users and support existing users more efficiently. Service providers want to reach as many users as possible with their mobile services. However, making services available on all mobile platforms is very costly due to the fragmentation of the mobile market. Developing native applications for each platform drastically increases the development costs. While native applications can fully exploit the features of a particular mobile platform, limited or no code can be shared between the different implementations. Each platform requires dedicated tools and different programming languages (e.g., Objective-C, C# and Java). Also, maintenance (e.g., updates or bug fixes) can be very costly. Hence, application developers are confronted with huge challenges. A promising alternative are mobile Cross-Platform Tools. A significant part of the code base is shared between the implementations for the different platforms. Further, many Cross-Platform Tools such as Cordova use client-side Web programming languages to implement the application logic, supporting programmers with a Web background.

Although several Cross-Platform Tools became more mature during the last few years, some skepticism towards CPTs remains. For many developers, the limited access to native device features (i.e., sensors and other platform APIs) remains an obstacle. In many cases, the developer is forced to use a limited set of the native APIs, or to use a work-around –which often involves native code– to achieve the desired functionality. This paper specifically tackles the use case of data storage APIs in Cordova. This paper extends our previous work [1] with an elaboration of the inner workings of the Cordova framework and new experiments concerning the employed bridge mechanisms.

Cordova is one of the most used CPTs [2][3]. It is a Web-to-native wrapper, allowing the developer to bundle Web apps into standalone applications.

Contribution. The contribution of this paper is fourfold. First, the Cordova framework including the Cordova Bridge is discussed. Second, four types of data storage strategies are distinguished in the setting of mobile applications (i.e., variables, databases, files and sensitive data). The support for each strategy using both native and Cordova development is analysed and compared. Third, based on this analysis a new Cordova plugin that extends the Cordova Storage API coverage is designed and developed. This plugin tackles a shortcoming in the currently available Cordova APIs. Finally, the security and performance of the different native and Cordova storage mechanisms is evaluated for both the Android and iOS platform.

The remainder of this paper is structured as follows. Section II points to related work. Section III gives an overview of CPTs. Section IV discusses the inner workings of Cordova applications, followed by Section V where an overview of data storage strategies and their API coverage in Cordova and native applications is given. The design and implementation of NativeStorage, a new Cordova storage plugin, is presented in Section VI. Section VII presents a security and performance evaluation of the Cordova and native storage mechanisms with a strong focus on the performance evaluation. The final section presents the conclusions and points to future work.

II. Related Work

Many studies compare CPTs based on a quantitative assessment. For instance, Rösler et al. [4] and Dalmasso et al. [5] evaluate the behavioral performance of cross-platform applications using parameters such as start-up time and memory consumption. Willocx et al. [6] extend this research and include more CPTs and criteria (e.g., CPU usage and battery usage) in the comparison. Further, Ciman and Gaggi [7] focus specifically on the energy consumption related to accessing sensors in cross-platform mobile applications. These studies are conducted using an implementation of the same application in a set of cross-platform tools and with the native development tools. This methodology provides useful insights in the overall performance overhead of using CPTs. Other research focuses on evaluating the performance of specific functional components. For instance, Zhuang et al. [8] evaluate the performance of the Cordova SQlite plugin for data storage. The work presented in this paper generalizes this work by providing an overview and performance analysis of the different data storage mechanisms available in Cordova, and comparing the performance with native components.

Several other studies focus on the evaluation of cross-platform tools based on qualitative criteria. For instance, Heitkötter et al. [9] use criteria such as development environment, maintainability, speed/cost of development and user-perceived application performance. The user-perceived performance is analyzed further in [10], based on user ratings and comments on cross-platform apps in the Google Play Store. The API coverage (e.g., geolocation and storage) of cross-platform tools is discussed in [11]. It is complementary with the work presented in this paper, which specifically focuses on the API coverage, performance and security related to data storage.

III. Cross-Platform Tools

Cross-platform tools can be classified in five categories based on their employed strategy [12]: JavaScript frameworks, Web-to-native wrappers, runtimes, source code translators and app factories. The latter provides a drag-and-drop interface to allow the creation of simple applications without programming knowledge. This paper focuses on the web-to-native wrapper Cordova. However, an overview of relevant CPT strategies is given to illustrate the differences, similarities and rationales. We can further classify cross-platform tool strategies in: (I) web-based CPTs and (II) source-code translators and runtimes.

A. Web-Based Cross-Platform Tool Strategies

JavaScript frameworks as well as Web-to-native wrappers make use of Web technologies for the development of mobile applications. A major advantage of these tools is that they enable Web developers to participate in mobile application development. Due to the availability of web browser capabilities in mobile operating systems this strategy is widely adopted. The user interface of such an application is developed with HTML and CSS, and the functionality is implemented using JavaScript.

To adapt to the specific interfaces and navigation patterns of mobile applications new mobile Javascript frameworks have been developed. These mobile interfaces are optimized for smaller screen sizes compared to regular websites and, for instance, provide support for the touch UI of mobile devices. Some frameworks also try to mimic the UI of native applications by providing native skins. Thereby tailoring the UI of the application to the look and feel of the platform on which it is running. These skins, however, do not provide a fully native experience. Most JavaScript frameworks also support traditional architectural design patterns such as Modelviewcontroller (MVC) and Modelviewviewmodel (MVVM) to facilitate the development of well-structured and maintainable code. Examples of such mobile JavaScript frameworks are: JQuery Mobile [13], Ionic [14] and Sencha Ext JS [15].

Two strategies can be applied to distribute the applications to end-users. First, an application (i.e., **Web app**) can be hosted on a Web server. This flexible approach allows the user to access the application in a mobile browser in a platform-agnostic manner. Hence, a large market can be easily reached and the time-to-market can be short. However, the application is constrained to the resources available in the browser. For instance, the JavaScript API of the browser (e.g., access to sensors such as accelerometer and GPS) is more limited than using native approaches. Furthermore, accessing the application is more cumbersome than starting an application installed on the device.

A second strategy is to pack the Web app into a standalone application by using a **Web-to-native wrapper**. The resulting application is often called a hybrid app due to the fact that it has both native as well as web characteristics. The packaging results in an application that can be submitted to the app stores of the different platforms. The Web app does not longer reside in the browser, but in a chromeless (without the window decoration of a regular web browser) WebView. The application consists of the WebView wrapper and the applications HTML, CSS and JavaScript files. The Web-to-native wrapper also features a JavaScript bridge that allows access to a broader range of platform APIs compared to the browser-exposed functionality. A typical structure of a hybrid application is illustrated in Figure 1. The most popular Web-to-native wrapper is Apache Cordova, formerly PhoneGap.

B. Source-Code Translators and Runtimes

Another CPT strategy is translating source code to code which can be understood by the underlaying platform such as a runtime. **Runtimes** shield applications from underlying platform differences through compatibility layers. The application source code can be either compiled or interpreted by the runtime during execution.

Also, the source code can be translated to the platform's native language or to executable byte code via a **Source-Code Translator**. Popular source code translators are Xamarin [16] and Qt [17]. In most cases, a combination of a source-code translator and a runtime is employed where a compilation step translates the source code to a binary or intermediary language that runs on the runtime. In a minority of tools, the source code will run straight on the runtime (e.g., Titanium [18]) or will be translated to native source code without a runtime (e.g., NeoMAD [19]). For each platform, the resulting source code is compiled using the development tools provided by the platform developer.

IV. Cordova Framework

A typical Cordova application consists of three important components: the application source, the WebView and plugins, as depicted in Figure 1. The Cordova framework allows these components to exchange information. A crucial component is the Cordova bridge which provides a way to connect the client-side native code with the JavaScript source.

A. Cordova Application Structure

The application code is loaded in a chromeless WebView. By default, Cordova applications use the WebView bundled

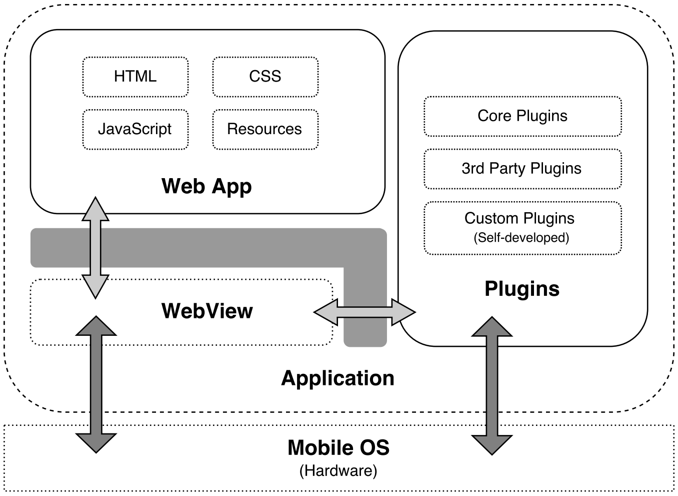


Figure 1. Structure of a Cordova application. Light grey arrows represent JavaScript calls, darker grey arrows represent native calls. The Cordova framework is illustrated by the grey area.

with the operating system. An alternative is to include the **Crosswalk WebView** [20]. The Crosswalk WebView provides uniform behaviour and interfaces between different (versions of) operating systems. Hence, developers do not need to take into account the differences in behaviour/APIs between the WebViews contained in the different (versions of) operating systems. Despite the efforts of the Crosswalk project, the maintenance and further developments have halted. The Android platform WebView has moved to an application so it can be updated separately to the Android platform. As a consequence, the crosswalk WebView will not yield a better performance on devices were the WebView is kept up-to-date. Therefore, we do not consider the impact of the crosswalk project on the performance of data storage strategies in this paper.

Cordova developers have two options for accessing device resources: the HTML5 APIs provided by the WebView and plugins. Despite the continuously growing HTML5 functionality [21] and the introduction of Progressive Web Apps [22], the JavaScript APIs provided by the WebView are not –yet– sufficient for the majority of applications. They do not provide full access to the diverse resources of the mobile device, such as sensors (e.g., accelerometer, gyroscope) and functionality provided by other applications installed on the device (e.g., contacts, maps, Facebook login). Plugins allow JavaScript code to access native APIs by using a JavaScript bridge between the Web code and the underlying operating system. Commonly used functionality such as GPS are provided by Cordova as *core plugins*. Additional functionality is provided by over 1000 third-party plugins, which are freely available in the Cordova plugin store [23]. Plugins consist of both JavaScript code and native code (i.e., Java for Android, Objective-C and recently Swift for iOS). The JavaScript code provides the interface to the developer. The native source code implements the functionality of the plugin and is compiled when building the application. The Cordova framework provides the JavaScript bridge that enables communication between JavaScript and native components. For each platform, Cordova supports several bridging mechanisms. At runtime, Cordova selects a bridging mechanism. When an error occurs, it switches to another mechanism. Independently of the selected bridging mechanism, the data requires several conversion steps before and after crossing the bridge.

B. Cordova Bridges

It is crucial to understand the inner workings of this bridge to be able to correctly evaluate the performance of the Cordova storage APIs. Here we consider the used bridge techniques on the Android and iOS platform.

Default in Android, Cordova uses the `addJavaScriptInterface` method for reaching the native side. This method injects a supplied object into the WebView. Afterwards an interface to the client-side Android code is accessible in JavaScript running in the WebView. This results in a bridge which can be invoked in JavaScript. Cordova invokes the `evaluateJavascript` method to execute JavaScript code in client-side Android.

The following bridge techniques are used in Android to exchange data and commands:

- JavaScript to Native Android:
  - **JS Object** (default). Methods of a Java object are directly accessed in Javascript via the `addJavaScriptInterface` as described above.
  - **Prompt**. The data is communicated through the prompt functionality of the WebView. This is used pre-JellyBean (i.e., Android 4.1), where `addJavascriptInterface` is disabled.
- Native Android to Javascript:
  - **Evaluation Bridge** (default). Through `evaluateJavascript` native Android code can execute JavaScript directly. This bridge was recently added and is now the default bridge.
  - **Polling**. The JavaScript side can be accessed by periodically polling for messages using the Javascript to Native Bridge.
  - **Event interception**. The Javascript code intercepts events (i.e., Load URL and Online event) and extracts the message from that event.

In iOS similar bridge techniques can be used. The javascript-to-native bridge is realized via URL loading interposition. On the JavaScript side a URL is loaded in an iframe or via XMLHttpRequest (XHR). This loading is intercepted by the native side. The JavaScript side communicates via the native side by encoding messages inside these URLs. The native side can access the JavaScript side by executing JavaScript code via the WebView's method `StringByEvaluatingJavaScriptFromString`.

V. Data Storage In Cordova

This work focuses on data storage mechanisms in Cordova applications. Four types of data storage strategies are distinguished: files, databases, persistent variables and sensitive data. Databases are used to store multiple objects of the same structure. Besides data storage, databases also provide methods to conveniently search and manipulate records. File storage can be used to store a diverse set of information such as audio, video and binary data. Persistent variables are stored as key-value pairs. It is often used to store settings and preferences. Sensitive data (e.g., passwords, keys, certificates) are typically handled separately from other types of data. Mobile operating systems provide dedicated mechanisms that increase the security of sensitive data storage.

The remainder of this section discusses the storage APIs available in Cordova and native Android/iOS. A summary of the results is shown in Table I.

TABLE I. STORAGE API COVERAGE

	Cordova	Android	iOS
Databases	WebSQL IndexedDB SQLite Plugin	SQLite	SQLite
Files	Cordova File Plugin	java.io	NSData
Persistent Variables	LocalStorage	Shared Prefs	NSUserDefaults Property Lists
Sensitive Data	SecureStorage Plugin	KeyStore Keychain	Keychain

A. Databases

Android and iOS provide a native interface for the **SQLite** library. Cordova supports several mechanisms to access database functionality from the application. First, the developer can use the database interface provided by the WebView. Both the native and CrossWalk WebViews provide two types of database APIs: **WebSQL** and **IndexedDB**. Although WebSQL is still commonly used, it is officially deprecated and thus no longer actively supported [24]. Second, developers can access the native database APIs via the **SQLite Plugin** [25].

B. Files

In Android, the file storage API is provided by the **java.io** package, in iOS this is included in **NSData**. Cordova provides a core plugin for File operation, namely **Cordova File Plugin** (cordova-plugin-file) [26]. Files are referenced via URLs which support using platform-independent references such as *application_folder*.

C. Persistent Variables

In Android, storing and accessing persistent variables is supported via **SharedPreferences**. It allows developers to store primitive data types (e.g., booleans, integers, strings). iOS developers have two options to store persistent variables: **NSUserDefaults** and **Property Lists**. NSUserDefaults has a similar behaviour to SharedPreferences in Android. Property Lists offer more flexibility by allowing storage of more complex data structures and specification of the storage location. Cordova applications can use the **LocalStorage** API provided by the Android and iOS WebView. Although it provides a simple API, developers should be aware of several disadvantages. First, LocalStorage only supports storage of strings. More complex data structures need to be serialized and deserialized by the developer. Second, LocalStorage is known [27] to perform poorly on large data sets and has a maximum storage capacity of 5MB.

```
// coarse grained API
NativeStorage.setItem("reference_to_value",<value>,
    <success-callback>, <error-callback>);
NativeStorage.getItem("reference_to_value",<success-
    callback>, <error-callback>);
NativeStorage.remove("reference_to_value",<success-
    callback>, <error-callback>);
NativeStorage.clear(<success-callback>, <error-
    callback>);
```

Listing 1. NativeStorage – Coarse-grained API

```
// fine grained API
NativeStorage.put<type>("reference_to_value",<value>,
     <success-callback>, <error-callback>);
NativeStorage.get<type>("reference_to_value",<
    success-callback>, <error-callback>);
NativeStorage.remove("reference_to_value",<success-
    callback>, <error-callback>);
```

Listing 2. NativeStorage – Fine-grained API

D. Sensitive Data

Android provides two mechanisms to store credentials: the **KeyChain** and the **KeyStore**. A KeyStore is bound to one specific application. Applications can not access credentials in KeyStores bound to other applications. If credentials need to be shared between applications, the KeyChain should be used. The user is asked for permission when an application attempts to access credentials in the KeyChain. Credential storage on iOS is provided by the **Keychain**. Credentials added to the Keychain are, by default, app private, but can be shared between applications from the same publisher. Cordova developers can use the credential storage mechanisms provided by Android and iOS via the **SecureStorage** (cordova-plugin-secure-storage) [28] plugin.

VI. NATIVESTORAGE PLUGIN

An important limitation of using HTML5 APIs (e.g., IndexedDB and LocalStorage) to store data in Cordova applications is that both on Android and iOS the cache of the WebView can be cleared when, for instance, the system is low on memory. This section presents NativeStorage [29], a Cordova plugin for persistent data object storage, mitigating the limitations of the HTML5 storage mechanisms.

A. Plugin Requirements

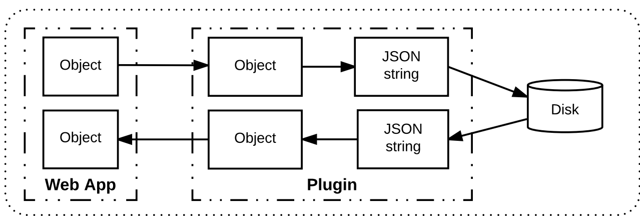
The requirements of the plugin are listed below:

- R_1 Persistent and sufficient storage
- R_2 Storage of both primitive data types and objects
- R_3 Support for Android and iOS
- R_4 App private storage
- R_5 Responsive APIs
- R_6 A user-friendly API

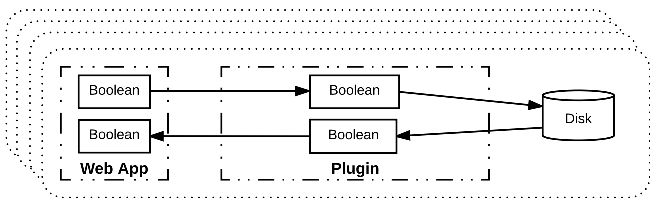
B. Realisation of NativeStorage

The plugin consists of JavaScript and native code. The JavaScript API provides the interface to application developers. The native side handles the storage of variables using native platform APIs.

NativeStorage provides two sets of JavaScript APIs, a fine-grained and a coarse-grained API, which are both asynchronous and non-blocking. The coarse grained API (Figure 2a) provides a type-independent interface, variables are automatically converted to JSON objects via the JSON interfaces provided by the WebView and passed as string variables to the native side. When a value is retrieved, the WebView is used to convert the string back to an object. The fine-grained API (Figure 2b) provides a separate implementation for the different JavaScript types. On the native side, the variables are stored via SharedPreferences in Android and NSUserDefaults in iOS.



(a) Coarse-grained API



(b) Fine-grained API

Figure 2. NativeStorage API

C. Evaluation of NativeStorage

The plugin is evaluated based on the previously listed requirements.

Persistent storage is provided via the native storage mechanisms. The documentation of the used native mechanisms does not state a limitation on the storage capacity. Hence, as opposed to LocalStorage, the storage capacity is only limited by the available memory on the device, satisfying R_1.

The native part of the plugin is developed for both Android and iOS. These mobile operating systems have a combined market share of 99% [30]. The used native storage mechanisms were introduced in iOS 2.0 and Android 1.0. The plugin, hence, provides support for virtually all versions of these platforms used in practice, satisfying R_3.

The plugin uses NSUserdefaults and SharedPreferences to store the data in app-private locations, ensuring that the variables can not be accessed from outside the application. This satisfies R_4.

The APIs are implemented using an asynchronous non-blocking strategy, facilitating the development of responsive applications (conform R_5).

Web developers are familiar with duck typing used in languages such as JavaScript. These types of languages often have APIs that don't distinguish between data types. The coarse-grained API provides such a storage mechanism. This API is shown in Listing 6. Not all Cordova developers, however, have a Web background. Therefore, a fine-grained API (Listing 5) is provided for developers who are more comfortable with a statically typed language, satisfying R_6 and R_2. Using both the coarse- and fine-grained API, the different JavaScript data types can be stored. Developers, however, need to be aware that the object storage relies on the JSON interface of the WebView to convert the object to a JSON string representation. The WebView, for instance, does not support the conversion of circular data structures. These types of objects, hence, need to be serialized by the developer before they can be stored.

Since its release to Github [31] and NPM [32] the plugin has been adopted by many Cordova application developers. We have registered over 16 500 downloads per month; with an overall number of downloads of 172 250 over a time span of two years.[1] Furthermore, the plugin is part of the 4% most downloaded packages on NPM. The plugin has been adopted in Ionic Native (Ionic 2) [33] and the Telerik plugin marketplace [34]. Telerik verifies that plugins are maintained and documented, thereby ensuring a certain quality.

VII. EVALUATION

The evaluation of the data storage mechanisms consists of two parts: a quantitative performance analysis and a security evaluation.

A. Performance

Developers want to be aware of the potential performance impact of using a CPT for mobile app development [12]. This section evaluates the performance of the different storage mechanisms for Cordova applications and compares the results with the native alternatives. Each storage strategy is tested by deploying a simple native and Cordova test application that intensively uses the selected storage strategy on an Android and iOS device. For Android the Nexus 6 running Android 6 was used, for iOS the IPhone 6 running iOS 9 was used. The test application communicates the test results via timing logs that are captured via Xcode for iOS and Android Studio for Android. The experiments were run sufficient times to ensure the measurements adequately reflect the performance of the tested storage mechanisms.

1) Databases

a) Test Application The database test application executes 300 basic CRUD operations (i.e., 100 x create, 100 x read, 50 x delete and 50 x read) of objects containing two string variables. The performance is determined by means of measuring the total duration of all the transactions. This test has been executed using the SQLite (native and Cordova), WebSQL (Cordova) and IndexedDB (Cordova) mechanisms.

b) Results and Comparison The results are presented in Table II. The mechanism for retrieving values by means of an index clearly results in a better performance compared to the SQL-based mechanisms. This analysis shows that IndexedDB provides an efficient way of storing and retrieving small objects. WebSQL –provided by the WebView– acts as a wrapper around SQLite. This is illustrated by the performance overhead associated with this mechanism. The deprecation of the specification/development stop could also have contributed to the performance penalty. The SQLite plugin suffers from a performance overhead caused by the interposition of the Cordova framework/bridge and has consequently a noticeable

[1]The statistics of the NativeStorage plugin can be found at https://npm-stat.com/charts.html?package=cordova-plugin-nativestorage.

performance overhead. The performance overhead introduced by the Cordova bridge is discussed in more detail in the following section.

TABLE II. RATIO OF DATABASE EXECUTION TIME TO THE NATIVE (SQLITE) OPERATION DURATION (IN %). *IN IOS INDEXEDDB IS ONLY SUPPORTED AS OF IOS 10.

	Android *Nexus 6*	iOS *iPhone 6*
SQLite (Native)	100	100
IndexedDB	6.94	12.47*
WebSQL	153	128
SQLite Plugin	133	116

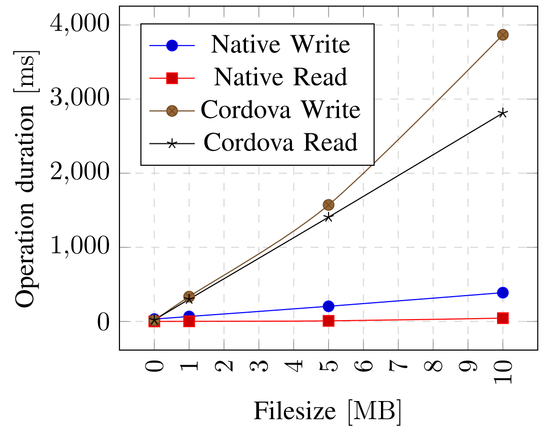


Figure 4. Duration of file operations in iOS

2) *Files*

a) Test Application The test application distinguishes between read and write operations. Each operation is tested using different file sizes, ranging from small files (~ 1 kB) to larger files (~ 10 MB). The performance of small files provides a baseline for file access. The performance of the read and write operations itself can be determined via the results of the large files. This test has been conducted ten times for each file size. The read and write operations consist of different steps on Android and iOS. Both the duration of the individual steps and the entire operation (i.e., read/write) is measured via timestamps. The application's memory footprint is measured via Instruments tool (Activity Monitor) in Xcode and via Memory Monitor in Android Studio.

b) Results and Comparison The results of the timing analysis on Android and iOS are presented in Figures 3 and 4, respectively. In both Android and iOS a significant performance difference between the native and the Cordova mechanism can be observed. R/W operations via the file plugin take longer compared to the native mechanisms. On top of a performance overhead, Cordova also comes with a higher memory consumption, especially in iOS (Figure 5).

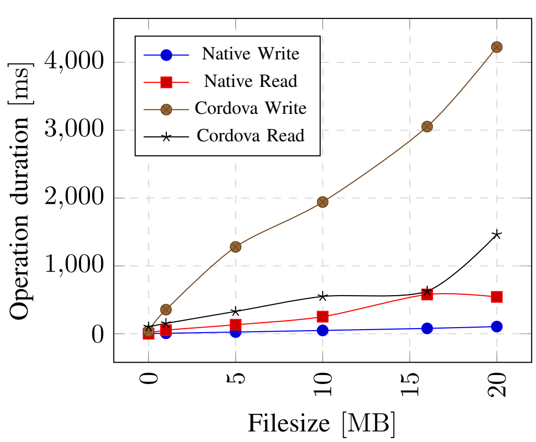


Figure 3. Duration of file operations in Android

Speed. Tables III and IV give a fine-grained overview of the different operations executed during respectively a file read and write using the Cordova platform on Android. Tables V and VI provide the results for iOS. Before data can be sent over the Cordova bridge, it needs to be converted to a string. This can create significant overhead when large binary files such as images need to be manipulated. Before they are sent over the bridge, the binary data is converted to a Base64 string. On Android, this is illustrated in the *Processing file* component of Table IV. Sending the data over the bridge also comprises a significant part of the overhead (i.e., *Sending over bridge*, from Table III). For small files, the overhead originates for the most part from resolving the platform-independent URL to a local path and retrieving meta-data. Similar observations can be made based on the iOS results.

TABLE III. EXECUTION TIME OF COMPONENTS ASSOCIATED WITH A READ OPERATION IN CORDOVA ANDROID (FILE PLUGIN). THE PROCEDURE "SENDING OVER BRIDGE" CONSISTS OF ENCODING, SENDING AND DECODING MESSAGES FROM THE JAVASCRIPT SIDE TO THE NATIVE SIDE.

Component	**Duration** [1 MB]		**Duration** [20 MB]	
	(ms)	(% total)	(ms)	(% total)
Resolve to local URL	58	**46**	59	7.56
Native reading	20	16	366	**47**
Sending over bridge	28	22	339	**43**
Total	126		780	

Memory. In iOS, applications manipulating large files will require large amounts of memory. This is illustrated in Figure 5. As shown, reading and writing a 10 MB file results in 400 MB of allocated memory. Reading and writing files larger than 10 MB can result in unstable behavior on iOS due to the large memory requirements. A solution for developers is to split large file operations in different steps.

c) Conclusion File storage on Apache Cordova comes with a number of limitations in terms of performance. This is a result of the Cordova framework/bridge technology. Allowing binary data to pass over the Cordova bridge could significantly improve the performance of plugins that perform operations on binary data. For instance, in [35] a bridging technology is presented that allows access to native device APIs in HTML5

TABLE IV. Execution time of components associated with a write operation in Cordova Android (File plugin). The procedure "processing file" converts the bytes –as an ArrayBuffer– to a string array. The "execute call delay" represents the delay between the write command executed in JavaScript and the execution at the native side.

Component	**Duration** [1 MB]		**Duration** [20 MB]	
	(ms)	(% total)	(ms)	(% total)
Processing file	108	**65**	1290	**56**
Execute call delay	38	**23**	632	**28**
Writing	20	12	369	16
Total	166		2291	

TABLE V. Performance read components in Cordova iOS

Component	**Duration** [1 MB]		**Duration** [10 MB]	
	(ms)	(% total)	(ms)	(% total)
Resolve to local URL	11	3.56	16	0.6
Native reading	13.98	4.52	70	2.47
Arguments to JSONArray	202.77	**65.62**	2037.93	**71.88**
Sending over bridge	59.93	**19.39**	587.19	**20.71**
Total	309		2835	

TABLE VI. Performance write components in Cordova iOS

Component	**Duration** [1 MB]		**Duration** [10 MB]	
	(ms)	(% total)	(ms)	(% total)
Processing file	266	**97**	2614	**96**
Native writing	7	3	96	4
Total	273		2710	

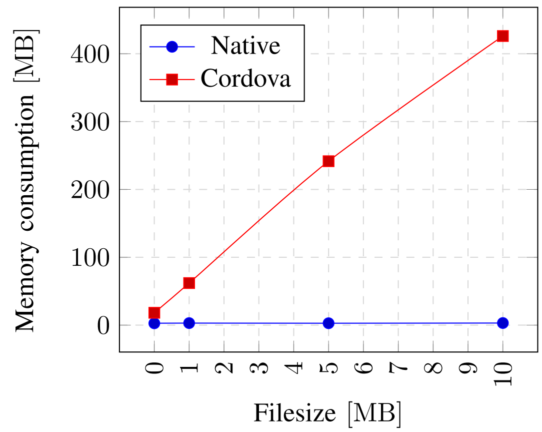


Figure 5. Memory consumption as a result of file operations in iOS

applications via WebSockets and HTTP servers, supporting the use of binary data.

3) Persistent variables

a) Test Application The performance is examined via storing and retrieving string values. The total duration of storing and retrieving a thousand variables is measured. The average storage and retrieval time is used to compare the different storage mechanisms. The Cordova mechanisms are LocalStorage and NativeStorage. These are compared to NSUserDefaults (iOS), Property Lists (iOS) and SharedPreferences (Android).

b) Results and Comparison All mechanisms have an execution time under 1 ms, with the exception of NativeStorage and Property Lists. The set operation takes around 1.9 ms, the get operation takes less than 1 ms. NativeStorage is the only mechanism which uses the Cordova bridge and framework, introducing a certain overhead. However, the NativeStorage API is asynchronous, hence, developers can continue processing while the value is being stored. The listed measurements include the time until the callback is fired. Property Lists load an entire file in an array, after which individual parameters can be read. As a consequence, the performance of the get operation, which takes 9.83 ms, is worse compared to the native alternatives. SharedPreferences and NSUserDefaults also load all parameters in memory, but this is done during the initialisation phase of the application, which is not incorporated in the measurements.

4) Cordova Bridges In addition to the evaluation of the data storage strategies in Cordova, compared to native, the performance of the utilized bridges are studied.

a) Test Application The performance of the bridges is tested by means of measuring the execution time of reading and writing different file sizes for each employed bridge. The used file sizes range from small files (1 kB) to larger files (10 MB). The test application reads and writes each file 1000 consecutively. The execution time for each transaction (i.e., read and write) is averaged over 1000 iterations. For larger files, such as $\sim$ 1 MB and $\sim$ 10 MB, the number of iterations is reduced to 100 and 20, respectively. We only consider the Android platform because the bridge techniques in iOS are rather limited.

First, the Cordova bridge allowing communication from the native to the JavaScript side (i.e., evaluation, online event, URL loading and polling) is studied. The bridge between JavaScript to native is unaltered, i.e., the default bridge is used. Second, the performance of the bridges connecting the JavaScript to the native side (i.e., JS object and prompt) is considered. For more information on the bridge mechanisms see Section VII-A4

The bridge experiments were conducted on a Samsung Galaxy S8 running Android 7.0.

b) Results and Conclusion The results [36], concerning the bridge connecting native to JavaScript, are presented in Tables VII and VIII. A first observation is that the polling mechanism could not be used as a bridge for the used Android version. In addition, the default bridge, denoted *Eval*, does not always surpass the other available bridges in terms of performance. This can be seen for reading as well as writing files. Depending on the file size the performance of other bridges are better than the default bridge. The same conclusion can be made when considering the bridges between the JavaScript side to the native side. However, the performance difference between these bridges is not as pronounced as with the native to JavaScript bridges. These results are added to this paper for the sake of completeness. They are presented in Table IX.

Based on the measurements, we can conclude that a single bridge technique is not appropriate for every use case when considering the performance. Hence, we propose a system were plugin developers can request a certain bridge technique based on their requirements. A fine-grained flexible selection of bridges will allow the performance of the application to increase. The impact on the performance when switching between bridges is not studied in this paper.

TABLE VII. PERFORMANCE NATIVE TO JAVASCRIPT BRIDGES IN ANDROID WHEN READING FILES

Data Size [bytes]	Eval (default) [ms]	Online Event [ms]	URL Load [ms]	Polling [ms]
1k	**2.31**	6.46	3.28	N.A.
10k	**6.22**	6.80	6.74	N.A.
100k	19.60	**8.20**	32.06	N.A.
1M	106.08	**43.42**	176.71	N.A.
10M	1308.33	**641.48**	1823	N.A.

Bold measurements indicate the best performance.

TABLE VIII. PERFORMANCE NATIVE TO JAVASCRIPT BRIDGES IN ANDROID WHEN WRITING FILES

Data Size [bytes]	Eval (default) [ms]	Online Event [ms]	URL Load [ms]	Polling [ms]
1k	**19.34**	43.03	23.34	N.A.
10k	36.74	42.85	**26.20**	N.A.
100k	42.60	40.74	**37.61**	N.A.
1M	424.65	419.67	**398.50**	N.A.
10M	4465.48	5996.29	**4398.43**	N.A.

Bold measurements indicate the best performance.

TABLE IX. PERFORMANCE JAVASCRIPT TO NATIVE BRIDGES IN ANDROID WHEN READING AND WRITING FILES

Data Size	Write		Read	
[bytes]	JS Object (default) [ms]	Prompt [ms]	JS Object (default) [ms]	Prompt [ms]
1k	21.88	22.02	2.73	3.17
10k	24.01	45.82	4.36	9.12
100k	46.19	43.51	21.38	19.80
1M	403.58	431.39	107.63	99.02
10M	5679.57	5606.48	1408.86	1406.48

B. Security

On both Android and iOS the security of storage mechanisms strongly depends on the storage location and the platform's backup mechanisms. Data stored inside the sandbox of the application is only accessible by the application. However, the backup mechanisms used in iOS and Android can result in the exposure of sensitive data [37, 38, 39], or potentially exhausting the limited cloud storage capacity. On iOS, this can result in the rejection of the application (conform the Data Storage Guidelines [40]). On Android, data stored inside the application sandbox (e.g., the WebView's storage) is included if a backup is taken. The Backup API of Android can be used to explicitly blacklist data that should not be backed up. On iOS, whether or not a file is included in the backup depends on the folder in which it is stored. For instance, by default, Cordova stores the WebView's data in a folder that allows backups. This behavior can, however, be changed by modifying a Cordova parameter.

1) Databases All database mechanisms are by default private to the application and can be backed up on both mobile platforms, with the exception of the SQLite plugin in iOS. The plugin initially followed the default behaviour, but as a security measure the default storage location of the plugin in iOS was changed to a directory which is not backed up. This SQLite plugin also has an encrypted alternative, i.e., **cordova-sqlcipher-adapter**. This alternative provides a native interface to SQLCipher, encrypting SQLite databases via a user-supplied password.

2) Files In iOS files are protected by a protection class. Each of these classes corresponds to different security properties. As of iOS 7, all files are by default encrypted individually until first user authentication. The file plugin doesn't allow changing this default behaviour. Native, each file can be secured using a protection class best suited for the security requirements of that file. The plugin allows the developer to choose between folders that are public/private and backup-enabled/disabled. However, on Android backup-disabled locations can be accessed by other applications.

3) Persistent variables All persistent variable storage mechanisms are private to the application and included in backups on both mobile platforms, with the exception of Property List. Property lists can be stored in arbitrary locations, and can be backed up depending on the specified location.

4) Sensitive Data The Secure Storage plugin provides storage of sensitive data on Android and iOS. On iOS, the plugin uses the SAMKeychain [41] plugin which provides an API for the native iOS Keychain. The plugin allows app-global static configuration of the KeyChain items' accessibility. This could entail a security risk, as it does not allow fine-grained protection of individual items. When a user backs up iPhone data, the Keychain data is backed up but the secrets in the Keychain remain encrypted with a phone-specific key in the backup. The Android KeyChain only allows storage of private keys. Hence, for storing other tokens such as passwords or JWT tokens, an additional encryption layer is used. The plugin generates a key that is stored in the KeyChain and used to encrypt/decrypt sensitive data. The KeyChain on Android is not included in backups.

VIII. CONCLUSION AND FUTURE WORK

This paper presented an assessment of data storage strategies using the mobile cross-platform tool Cordova. An in-depth analysis was performed on the API coverage of the available data storage mechanisms in Cordova and Native applications. Based on the analysis, an additional Cordova storage plugin was developed that improves the storage of persistent variables.

Furthermore, the performance and security of the available storage mechanisms were evaluated. Our performance analysis shows that using the Cordova bridge comes with a significant performance penalty. Hence, the WebView's JavaScript API should be used when possible. Moreover, we also demonstrated that the default bridges used in Cordova do not always outperform the non-default bridges. However, apart from performance, other parameters such as functionality and security can have an impact on the selection of the storage mechanism.

Databases. If access to a full fledged SQL database is required, the SQLite plugin should be used. However, in most mobile applications, the functionality provided by the significantly faster IndexedDB interface of the WebView is sufficient.

Variables. As described in Sections VI and VII, it is recommended to use NativeStorage for storing persistent variables, since LocalStorage does not guarantee persistence over longer periods of time. This type of storage is often used to store preferences. Preferences are typically only accessed once or twice during the life cycle of the application. Hence, the performance overhead of NativeStorage does not have a significant impact on the performance of the application.

Files. The WebView does not provide a file storage API. Hence, developers have to use the core plugin, Cordova File Plugin (`cordova-plugin-file`).

Sensitive data. The security analysis presented in Section VII-B shows that plugins such as SecureStorage offer increased security compared to the WebView's JavaScript API because they benefit from the platform's native secure storage APIs. It is therefore recommended to use a plugin such as SecureStorage to store sensitive data.

Future work on this topic can include an enhancement of the Cordova framework where a fine-grained selection of bridge techniques is allowed. Thereby, improving the performance of Cordova applications. Furthermore, more CPTs can be included in the assessment of data storage strategies.

References

[1] G. Callebaut, M. Willocx, J. Vossaert, V. Naessens, and L. D. Strycker, "Assessment of data storage strategies using the mobile cross-platform tool cordova," in *MOBILITY 2017, The Seventh International Conference on Mobile Services, Resources, and Users*, J. Noll and K. El-Khatib, Eds., 2017, pp. 25–32.

[2] VisionMobile. Cross–Platform Tools 2015. Access date: 13/04/2016. [Online]. Available: http://www.visionmobile.com/product/cross-platform-tools-2015/

[3] ——. (2016) Developer economics state of the developer nation q1 2016. Access date: 13/04/2016. [Online]. Available: http://www.visionmobile.com/product/developer-economics-state-of-developer-nation-q1-2016/

[4] F. Rösler, A. Nitze, and A. Schmietendorf, "Towards a mobile application performance benchmark," in *International Conference on Internet and Web Applications and Services*, vol. 9, 2014, pp. 55–59.

[5] I. Dalmasso, S. K. Datta, C. Bonnet, and N. Nikaein, "Survey, comparison and evaluation of cross platform mobile application development tools," in *Wireless Communications and Mobile Computing Conference (IWCMC), 2013 9th International*. IEEE, 2013, pp. 323–328.

[6] M. Willocx, J. Vossaert, and V. Naessens, "Comparing performance parameters of mobile app development strategies," in *Proceedings of the International Workshop on Mobile Software Engineering and Systems*. ACM, 2016, pp. 38–47.

[7] M. Ciman and O. Gaggi, "Evaluating impact of cross-platform frameworks in energy consumption of mobile applications." in *WEBIST (1)*, 2014, pp. 423–431.

[8] Y. Zhuang, J. Baldwin, L. Antunna, Y. O. Yazir, S. Ganti, and Y. Coady, "Tradeoffs in cross platform solutions for mobile assistive technology," in *Communications, Computers and Signal Processing (PACRIM), 2013 IEEE Pacific Rim Conference on*. IEEE, 2013, pp. 330–335.

[9] H. Heitkötter, S. Hanschke, and T. A. Majchrzak, "Evaluating cross-platform development approaches for mobile applications," in *International Conference on Web Information Systems and Technologies*. Springer, 2012, pp. 120–138.

[10] I. Malavolta, S. Ruberto, T. Soru, and V. Terragni, "End users' perception of hybrid mobile apps in the google play store," in *2015 IEEE International Conference on Mobile Services*. IEEE, 2015, pp. 25–32.

[11] M. Palmieri, I. Singh, and A. Cicchetti, "Comparison of cross-platform mobile development tools," in *Intelligence in Next Generation Networks (ICIN), 2012 16th International Conference on*. IEEE, 2012, pp. 179–186.

[12] VisionMobile, "Cross-platform developer tools 2012, bridging the worlds of mobile apps and the web," *Februar y*, 2012, access date: 13/04/2016.

[13] JQuery Mobile. A Touch-Optimized Web Framework. Access date: 15/09/2017. [Online]. Available: https://jquerymobile.com/

[14] Ionic, "Hybrid vs. Native – An introduction to cross-platform hybrid development for architects and app development leaders," Tech. Rep. [Online]. Available: https://ionicframework.com/books/hybrid-vs-native

[15] Sencha. Sencha Ext JS. Access date: 15/09/2017. [Online]. Available: https://www.sencha.com/products/extjs

[16] Xamarin – Mobile App Development & App Creation Software. Access date: 29/11/2017. [Online]. Available: www.xamarin.com

[17] Qt – Cross-Platform software development for embedded & desktop. Access date: 29/11/2017. [Online]. Available: www.qt.io

[18] Appcelerator Inc. Appcelerator. Access date: 15/09/2017. [Online]. Available: http://www.appcelerator.com/

[19] NeoMAD – Cross-platform mobile development. Access date: 15/09/2017. [Online]. Available: http://neomades.com

[20] Crosswalk – Build world class hybrid apps. Access date: 29/05/2018. [Online]. Available: https://crosswalk-project.org

[21] A. Deveria and L. Schoors. Can I use ... ? Access date: 29/05/2018. [Online]. Available: https://caniuse.com/#search=HTML5

[22] Google. Progressive Web Apps. Access date: 29/05/2018. [Online]. Available: https://developers.google.com/web/progressive-web-apps/

[23] Apache Cordova. Cordova Plugins. Access date: 29/05/2018. [Online]. Available: https://cordova.apache.org/plugins/

[24] Web SQL Database documentation. Access date: 29/05/2018. [Online]. Available: https://dev.w3.org/html5/webdatabase/

[25] SQLite Plugin NPM website. Access date: 29/05/2018.

[Online]. Available: https://www.npmjs.com/package/cordova-sqlite-storage

[26] Cordova File Plugin NPM website. Access date: 29/05/2018. [Online]. Available: https://www.npmjs.com/package/cordova-plugin-file

[27] Cordova Storage documentation. Access date: 29/05/2018. [Online]. Available: https://cordova.apache.org/docs/en/latest/cordova/storage/storage.html

[28] SecureStorage Plugin NPM website. Access date: 29/05/2018. [Online]. Available: https://www.npmjs.com/package/cordova-plugin-secure-storage

[29] G. Callebaut and A. Rajiv, "NativeStorage – A Cordova Plugin," https://github.com/TheCocoaProject/cordova-plugin-nativestorage.

[30] "Smartphone os market share, q2 2016," http://www.idc.com/prodserv/smartphone-os-market-share.jsp, 2015, access date: 20/10/2016.

[31] (2016) Cordova plugin NativeStorage. [Online]. Available: https://github.com/TheCocoaProject/cordova-plugin-nativestorage

[32] NativeStorage Plugin NPM website. Access date: 29/05/2018. [Online]. Available: https://www.npmjs.com/package/cordova-plugin-nativestorage

[33] NativeStorage in the Ionic Framework documentation. Access date: 29/05/2018. [Online]. Available: http://ionicframework.com/docs/v2/native/nativestorage/

[34] Cordova Plugins in the Telerik Marketplace. Access date: 29/05/2018. [Online]. Available: http://plugins.telerik.com/cordova

[35] A. Puder, N. Tillmann, and M. Moskal, "Exposing native device apis to web apps," in *Proceedings of the 1st International Conference on Mobile Software Engineering and Systems*, ser. MOBILESoft 2014. New York, NY, USA: ACM, 2014, pp. 18–26. [Online]. Available: http://doi.acm.org/10.1145/2593902.2593908

[36] G. Callebaut. (2017) Performance results bridges in cordova (android). [Online]. Available: http://dx.doi.org/10.17632/kyxc59tfmv.1

[37] P. Teufl, T. Zefferer, and C. Stromberger, "Mobile device encryption systems," in *28th IFIP TC-11 SEC 2013 International Information Security and Privacy Conference*, 2013, pp. 203 – 216.

[38] P. Teufl, A. G. Fitzek, D. Hein, A. Marsalek, A. Oprisnik, and T. Zefferer, "Android encryption systems," in *International Conference on Privacy & Security in Mobile Systems*, 2014, in press.

[39] P. Teufl, T. Zefferer, C. Stromberger, and C. Hechenblaikner, "ios encryption systems - deploying ios devices in security-critical environments," in *SECRYPT*, 2013, pp. 170 – 182.

[40] iOS Data Storage Guidelines. Access date: 29/05/2018. [Online]. Available: https://developer.apple.com/icloud/documentation/data-storage/index.html

[41] S. Soffes. SAMKeychain. Access date: 29/05/2018. [Online]. Available: https://github.com/soffes/SAMKeychain

Fuzzy Regression Model with Modified Kernel and its Application to a Set of Real Data

Kiyoshi Nagata

Faculty of Business Administration
Daito Bunka University
Tokyo 175-8571, Japan
Email: nagata@ic.daito.ac.jp

Michihiro Amagasa

Faculty of Business Administration
Hokkai Gakuen University
Sapporo 062-8605, Japan
Email: amagasa@ba.hokkai-s-u.ac.jp

***Abstract*—Regression model is a popular and powerful model for finding a rule from large amount of collected data. It is widely used in various areas for predicting the value derived from observable values. Especially in multivariate numerical analysis, several types of regression models, not only linear but also polynomial or exponential, are established. In case of non-numerical data, although fuzzy regression models are proposed and investigated by some researchers, most of them are linear models. In order to construct a non-linear regression model with fuzzy type data set, new type of devices are needed since fuzzy numbers have a complicated behavior in multiplication and division. In this paper, we try to extend a linear fuzzy regression model to non-linear model by adapting a modified kernel method. Then, we apply the model of only low degree of polynomial kernel to a data set obtained by conducting questionnaire survey on purchasing decision making of electric assisted bicycle in Japan.**

Keywords–Fuzzy regression model; Kernel method; Decision making.

I. INTRODUCTION

As an analysis method of numerical big-data mining, the regression model is still playing an important role. However, the huge amount of data processing requires strong computing power and resources. In particular, when handling data with non-linear features, finding a proper regression model is not easy, sometimes even infeasible. The kernel method, so-called a kernel trick, is one of smart devices solving this kind of problem. A kernel defined on the product of a data set induces an element of Hilbert space, a space of functions with an inner product, and considering a linear model in the space gives us a non-linear model in the original space. Thus, only the calculation of kernels for the given data set is non-linear, and the calculation for solving the problem to give a model is performed in the linear operation method. The kernel method is applied to many analytical systems, such as the Principal Component Analysis (PCA) [20], the Canonical Correction Analysis (CCA) [8] [16], Fisher's Linear Discriminant Analysis (LDA) [17], the Support Vector Machine(SVM) [2] [10], the regression model [18] [21], etc.

In the real world, the collected data are sometimes expressed in linguistic values, and in order to apply well-known and authorized stochastic methods such as regression analysis, these values are transformed into numerical data. For instance, the price of a production or a service are determined from several factors, such as price of raw materials, selling expenses, consumer demand, etc. Also, the price has high correlations with the customer value of product or service. Gale [13, pp. 218-219] proposed a scenario where price satisfaction carries 40% of the weight and non-price attributes 60% in the customer-value equation, and showed a figure representing the relationship between relative performance overall score and relative price for luxury cars based on data. In that figure, the relative price is generically expressed in linguistic values such as “Higher”, “Lower”, etc., then these values are transformed into numerical values in order to plot corresponding points on the performance-price plane. For the price prediction model, Inoue et al. [15] proposed a sale price prediction model by fuzzy regression, and Michihiro Amagasa [4], also proposed a method to handle data with uncertainty in the model of regression analysis as an extension of their model. We also give a precise formulation of a multi-variable regression model where both explanatory variables and objective one are L-R type fuzzy numbers [1] [5].

Construction approaches for regression models handling fuzzy set are roughly divided into two types, one is Fuzzy Least Square Regression (FLSR) and the other is dual model for possibilistic regression. The concept of FLSR model is similar to that of ordinary regression model where each value of three vertexes is processed to minimize the sum of distances between the given data and the estimated values. D'Urso adopts this approach handling linear regression model with several types of input-output data, such as crisp-fuzzy, fuzzy-crisp, and fuzzy-fuzzy, with not only type1 fuzzy data set but also type2 fuzzy data set [24]. The dual model of possibilistic regression approach, originally proposed by Tanaka et al. [22] [23], gives upper and lower regression model by using linear programing analysis approach. Although their model is extended to non-linear models [14], explanatory variables are still crisp values. In this paper, we propose a non-linear regression model of fuzzy input-fuzzy output type as an extension of our previously proposed model in [5] by applying the kernel method.

The rest of the paper is organized as follows: In Section II, we will review general theory of the kernel method and give a concrete construction of quadratic and cubic kernels for a small number of variables. Section III is dedicated to a brief explanation of Guo and Tanaka's non-linear fuzzy regression model and the details of our linear model. Then, in Section IV, we describe the extension version of our model into non-linear type with modified kernels. Illustrative examples to see how the proposed model works are coming up, and application to a real data obtained by conducting a questionnaire survey on purchasing behavior of electric assisted bicycle in Japan is also described with some discussions. The last section, Section V, is the conclusion and the future works.

II. Kernel Theory

First, we give a brief description of kernel theory, then give an expression of the functions in the reproducing kernel Hilbert space for a quadratic kernel.

A. Overview of Kernel Theory

For any set $\mathcal{X}$ and the Hilbert space $\mathcal{H}$ of functions on $\mathcal{X}$ over $\mathbb{R}$, a positive definite kernel is a map

$$k : \mathcal{X} \times \mathcal{X} \to \mathcal{H}$$

satisfying

- $k(x, y) = k(y, x)$ for any $x,\ y \in \mathcal{X}$,
- For any $\{c_i\} \subset \mathbb{R}$ and any $\{x_i\} \subset \mathcal{X}$,

$$\sum c_i c_j k(x_i, x_j) \geq 0.$$

Here, we give some examples of kernel over $\mathbb{R}^k$.
For $\vec{x} = (x_1, \ldots, x_k), \vec{y} = (y_1, \ldots, y_k)$,

- $k(\vec{x}, \vec{y}) = \vec{x}^t\vec{y} = \sum_{i=1}^k x_i y_i$ (linear kernel)
- $k_P(\vec{x}, \vec{y}) = (\vec{x}^t\vec{y} + c)^d$,
  with $c \geq 0,\ 0 < d \in \mathbb{Z}$ (polynomial kernel)
- $k_E(\vec{x}, \vec{y}) = exp(\beta\vec{x}^t\vec{y})$, with $\beta > 0$ (exponential kernel)
- $k_G(\vec{x}, \vec{y}) = exp(-\frac{1}{2\sigma^2}\|\vec{x} - \vec{y}\|^2)$
  (Gaussian radial basis function kernel)
- $k_L(\vec{x}, \vec{y}) = exp(-\alpha\sum_{i=1}^k |x_i - y_i|)$ (Laplacian kernel)

If, for any $x \in \mathcal{X}$, there exists a function $k_x \in \mathcal{H}$ such that

$$f(x) =< f, k_x >_{\mathcal{H}}, (\forall f \in \mathcal{H}) \tag{1}$$

where $< \cdot, \cdot >_{\mathcal{H}}$ is the inner product of the Hilbert space, the Hilbert space $\mathcal{H}$ is called a Reproducing Kernel Hilbert Space (RKHS). It is shown that $k_x \in \mathcal{H}$ is unique, and $k(\cdot, x) = k_x$ is a positive definite kernel on $\mathcal{X}$ called the reproducing kernel.

Conversely, the following theorem is known in [6].

Theorem 1. *(Moore-Aronszajn) For any positive definite kernel on $\mathcal{X}$, there exist unique Hilbert space $\mathcal{H}$ satisfying*

1) *$k(\cdot, x) \in \mathcal{H}$ (for any $x \in \mathcal{X}$),*
2) *The subspace spanned by $\{k(\cdot, x); x \in \mathcal{X}\}$ is dense in $\mathcal{H}$,*
3) *k is the reproducing kernel of $\mathcal{H}$.*

Although Hilbert space has infinity dimension, solution of some optimization problem with data, if there is any, can be expressed as a linear combination of at most the number of data elements in $\mathcal{H}$. This is guaranteed by the following theorem in [19].

Theorem 2. *(The Representer Theorem) Let k be a kernel on $\mathcal{X}$ and let $\mathcal{H}$ be its associated RKHS. Fix $x_1, \cdots, x_n \in \mathcal{X}$, and consider the optimization problem*

$$\min_{f \in \mathcal{H}} D(f(x_1), \ldots, f(x_n)) + P(\|f\|^2_{\mathcal{H}}) \tag{2}$$

where P is nondecreasing and D depends only on $f(x_1), \cdots, f(x_n)$. If there is a minimizer, then it has the form of

$$f = \sum_{i=1}^n a_i k(\cdot, x_i) \tag{3}$$

with some $a_1, \cdots, a_n \in \mathbb{R}$. Furthermore, if P is strictly increasing, then every solution has this form.

B. Example Expression of RKHS Basis

From the representer theorem, we can express an optimal function as in the form of (3). However, if the given data set is big, we will have many unknown variables $\{a_i\}_{i=1,\ldots,n}$ to be determined. For the convenience of calculation, we try to reduce the number of components for the polynomial kernel and give an example for the quadratic polynomial kernel of tow cases, one is the case of $d = 2$ with $k = 3$ variables and the other is the case of $d = 2$, 3 with $k = 4$ variables.

From the representer theorem and the equation below,

$$k_P(\vec{x}, \vec{y}) = (\textstyle\sum_{j=1}^k x_j y_j + c)^d = \displaystyle\sum_{\substack{0 \leq e_1 + \cdots + e_k \leq d \\ 0 \leq e_j}} c^{d-(e_1+\cdot+e_k)} x_1^{e_1} \cdots x_k^{e_k} y_1^{e_1} \cdots y_k^{e_k}$$

we have that for any $(e_1, \ldots, e_k)$ such that $0 \leq e_1 + \cdots + e_k \leq d, 0 \leq e_i$, there exist $N =_{k+d} C_d$ vectors, $\vec{x}_1, \ldots \vec{x}_N$, and $a_1, \ldots, a_N$ satisfying

$$\sum_{i=1}^N a_i x_{i1}^{f_1} \cdots x_{ik}^{f_k} = \begin{cases} c^{-(d-(e_1+\cdot+e_k))} & \text{if}(f_1 \ldots, f_k) = (e_1, \ldots, e_k), \\ 0 & \text{otherwise.} \end{cases} \tag{4}$$

1) In case of $d = 2$ with $k = 3$: In a simple case of $d = 2$ with $k = 3$ variables, $N =_5 C_2 = 10$ and the left side of equation (4) is expressed as

$$\begin{pmatrix} x_{11}^2 & x_{21}^2 & \cdots & x_{10\,1}^2 \\ x_{12}^2 & x_{22}^2 & \cdots & x_{10\,2}^2 \\ x_{13}^2 & x_{23}^2 & \cdots & x_{10\,3}^2 \\ x_{11} & x_{21} & \cdots & x_{10\,1} \\ x_{12} & x_{22} & \cdots & x_{10\,2} \\ x_{13} & x_{23} & \cdots & x_{10\,3} \\ x_{11}x_{12} & x_{21}x_{22} & \cdots & x_{10\,1}x_{10\,2} \\ x_{11}x_{13} & x_{21}x_{23} & \cdots & x_{10\,1}x_{10\,3} \\ x_{12}x_{13} & x_{22}x_{23} & \cdots & x_{10\,2}x_{10\,3} \\ 1 & 1 & \cdots & 1 \end{pmatrix} \begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ \vdots \\ a_{10} \end{pmatrix}.$$

However, we only have to determine $\vec{x}_1, \vec{x}_2, \vec{x}_3$ and solve the 10 equations of (4) shown as follows.

$$\begin{pmatrix} x_{1j}^2 & x_{2j}^2 & x_{3j}^2 \\ x_{1j} & x_{2j} & x_{3j} \\ 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} a_1 \\ a_2 \\ a_3 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ c \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ c^2 \end{pmatrix},$$

or

$$\begin{pmatrix} x_{1j}^2 & x_{2j}^2 & x_{3j}^2 \\ x_{1j} & x_{2j} & x_{3j} \\ x_{1j}x_{1l} & x_{2j}x_{2l} & x_{3j}x_{3l} \end{pmatrix} \begin{pmatrix} a_1 \\ a_2 \\ a_3 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix},$$

where $j, l = 1, 2, 3$ and $j \neq l$. Just analyzing the invertibility of these matrices, we have 10 functions spanning the dense subspace $\mathcal{H}'_k$ of $\mathcal{H}_k$.

$$\mathcal{H}'_k =< k(\cdot, \vec{x}_i); i = 1, \ldots, 10 >_{\mathbb{R}},$$

where
$\vec{x}_1 = (1,0,0)$, $\vec{x}_2 = (0,1,0)$, $\vec{x}_3 = (0,0,1)$,
$\vec{x}_4 = (-1,0,0)$, $\vec{x}_5 = (0,-1,0)$, $\vec{x}_6 = (0,0,-1)$,
$\vec{x}_7 = (1,1,0)$, $\vec{x}_8 = (0,1,1)$, $\vec{x}_9 = (1,0,1)$, and
$\vec{x}_{10} = (0,0,0)$.

2) In case of $d = 2, 3$ with $k = 4$: For the analysis of the real data, we need to have base vectors which have $k = 4$ elements. We find out the basis for $d = 2$ and $d = 3$ with $N = {}_6C_2 = 15$ and $N = {}_7C_3 = 35$, respectively by considering the matrix expression of the left side of equation (4) expressed as

$$\begin{pmatrix} x_{11}^2 & x_{21}^2 & \cdots & x_{15\,1}^2 \\ \vdots & \vdots & \vdots & \vdots \\ x_{14}^2 & x_{24}^2 & \cdots & x_{15\,4}^2 \\ x_{11} & x_{21} & \vdots & x_{15\,1} \\ \vdots & \vdots & \vdots & \vdots \\ x_{14} & x_{24} & \cdots & x_{15\,4} \\ x_{11}x_{12} & x_{21}x_{22} & \cdots & x_{15\,1}x_{15\,2} \\ \vdots & \vdots & \vdots & \vdots \\ x_{13}x_{14} & x_{23}x_{24} & \cdots & x_{15\,3}x_{15\,4} \\ 1 & 1 & \cdots & 1 \end{pmatrix} \begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ \vdots \\ a_{15} \end{pmatrix}.$$

and

$$\begin{pmatrix} x_{11}^3 & x_{21}^3 & \cdots & x_{35\,1}^3 \\ \vdots & \vdots & \vdots & \vdots \\ x_{14}^3 & x_{24}^3 & \cdots & x_{35\,4}^3 \\ \vdots & \vdots & \vdots & \vdots \\ x_{11} & x_{21} & \cdots & x_{35\,1} \\ \vdots & \vdots & \vdots & \vdots \\ x_{14} & x_{24} & \cdots & x_{35\,4} \\ x_{11}^2x_{12} & x_{21}^2x_{22} & \cdots & x_{35\,1}^2x_{35\,2} \\ x_{11}^2x_{13} & x_{21}^2x_{23} & \cdots & x_{35\,1}^2x_{35\,3} \\ x_{11}^2x_{14} & x_{21}^2x_{24} & \cdots & x_{35\,1}^2x_{35\,4} \\ x_{12}^2x_{13} & x_{22}^2x_{23} & \cdots & x_{35\,2}^2x_{35\,3} \\ x_{12}^2x_{14} & x_{22}^2x_{24} & \cdots & x_{35\,2}^2x_{35\,4} \\ x_{13}^2x_{14} & x_{23}^2x_{24} & \cdots & x_{35\,3}^2x_{35\,4} \\ x_{11}x_{12}^2 & x_{21}x_{22}^2 & \cdots & x_{35\,1}x_{35\,2}^2 \\ x_{11}x_{13}^2 & x_{21}x_{23}^2 & \cdots & x_{35\,1}x_{35\,3}^2 \\ x_{11}x_{14}^2 & x_{21}x_{24}^2 & \cdots & x_{35\,1}x_{35\,4}^2 \\ x_{12}x_{13}^2 & x_{22}x_{23}^2 & \cdots & x_{35\,2}x_{35\,3}^2 \\ x_{12}x_{14}^2 & x_{22}x_{24}^2 & \cdots & x_{35\,2}x_{35\,4}^2 \\ x_{13}x_{14}^2 & x_{23}x_{24}^2 & \cdots & x_{35\,3}x_{35\,4}^2 \\ x_{11}x_{12} & x_{21}x_{22} & \cdots & x_{35\,1}x_{35\,2} \\ x_{11}x_{13} & x_{21}x_{23} & \cdots & x_{35\,1}x_{35\,3} \\ x_{11}x_{14} & x_{21}x_{24} & \cdots & x_{35\,1}x_{35\,4} \\ x_{12}x_{13} & x_{22}x_{23} & \cdots & x_{35\,2}x_{35\,3} \\ x_{12}x_{14} & x_{22}x_{24} & \cdots & x_{35\,2}x_{35\,4} \\ x_{13}x_{14} & x_{23}x_{24} & \cdots & x_{35\,3}x_{35\,4} \\ x_{11}x_{12}x_{13} & x_{21}x_{22}x_{23} & \cdots & x_{35\,1}x_{35\,2}x_{35\,3} \\ x_{11}x_{12}x_{14} & x_{21}x_{22}x_{24} & \cdots & x_{35\,1}x_{35\,2}x_{35\,4} \\ x_{11}x_{13}x_{14} & x_{21}x_{23}x_{24} & \cdots & x_{35\,1}x_{35\,3}x_{35\,4} \\ x_{12}x_{13}x_{14} & x_{22}x_{23}x_{24} & \cdots & x_{35\,2}x_{35\,3}x_{35\,4} \\ 1 & 1 & \cdots & 1 \end{pmatrix} \begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ \vdots \\ a_{15} \end{pmatrix}.$$

Analyzing the invertibility of these matrices, we have following sets of 15 and 35 vectors which induce the functions spanning the dense subspace $\mathcal{H}'_k$ of $\mathcal{H}_k$. For $d = 2$,

$$\mathcal{H}'_k =< k(\cdot, \vec{x}_i); i = 1, \ldots, 15 >_{\mathbb{R}},$$

with $\vec{x}_1 = (1,0,0,0)$, $\vec{x}_2 = (0,1,0,0)$, $\vec{x}_3 = (0,0,1,0)$,
$\vec{x}_4 = (0,0,0,1)$, $\vec{x}_5 = (-1,0,0,0)$, $\vec{x}_6 = (0,-1,0,0)$,
$\vec{x}_7 = (0,0,-1,0)$, $\vec{x}_8 = (0,0,0,-1)$, $\vec{x}_9 = (1,1,0,0)$,
$\vec{x}_{10} = (1,0,1,0)$, $\vec{x}_{11} = (1,0,0,1)$, $\vec{x}_{12} = (0,1,1,0)$,
$\vec{x}_{13} = (0,1,0,1)$, $\vec{x}_{14} = (0,0,1,1)$, and $\vec{x}_{15} = (0,0,0,0)$.

For $d = 3$,

$$\mathcal{H}'_k =< k(\cdot, \vec{x}_i); i = 1, \ldots, 35 >_{\mathbb{R}},$$

with $\vec{x}_1 = (1,0,0,0)$, $\vec{x}_2 = (0,1,0,0)$, $\vec{x}_3 = (0,0,1,0)$,
$\vec{x}_4 = (0,0,0,1)$, $\vec{x}_5 = (-1,0,0,0)$, $\vec{x}_6 = (0,-1,0,0)$,
$\vec{x}_7 = (0,0,-1,0)$, $\vec{x}_8 = (0,0,0,-1)$, $\vec{x}_9 = (2,0,0,0)$,
$\vec{x}_{10} = (0,2,0,0)$, $\vec{x}_{11} = (0,0,2,0)$, $\vec{x}_{12} = (0,0,0,2)$,
$\vec{x}_{13} = (1,1,0,0)$, $\vec{x}_{14} = (1,0,1,0)$, $\vec{x}_{15} = (1,0,0,1)$,
$\vec{x}_{16} = (0,1,1,0)$, $\vec{x}_{17} = (0,1,0,1)$, $\vec{x}_{18} = (0,0,1,1)$,
$\vec{x}_{19} = (-1,-1,0,0)$, $\vec{x}_{20} = (-1,0,-1,0)$,
$\vec{x}_{21} = (-1,0,0,-1)$, $\vec{x}_{22} = (0,-1,-1,0)$,
$\vec{x}_{23} = (0,-1,0,-1)$, $\vec{x}_{24} = (0,0,-1,-1)$,
$\vec{x}_{25} = (1,-1,0,0)$, $\vec{x}_{26} = (1,0,-1,0)$, $\vec{x}_{27} = (1,0,0,-1)$,
$\vec{x}_{28} = (0,1,-1,0)$, $\vec{x}_{29} = (0,1,0,-1)$, $\vec{x}_{30} = (0,0,1,-1)$,
$\vec{x}_{31} = (1,1,1,0)$, $\vec{x}_{32} = (1,1,0,1)$, $\vec{x}_{33} = (1,0,1,1)$,
$\vec{x}_{34} = (0,1,1,1)$, and $\vec{x}_{35} = (0,0,0,0)$.

III. Some Existing Fuzzy Regression Model

In this section, we will give a brief explanation of two fuzzy regression models, one is crisp-input and fuzzy-output type by Guo and Tanaka, and the other is fuzzy-input and fuzzy-output type.

A. Guo and Tanaka's Non-Linear Model

Guo and Tanaka have investigated the dual possibilistic regression models of both linear and non-linear types with crisp-input and symmetric triangular fuzzy-output in [14]. At first, the linear model whose output $Y = (y; p)_F = (y; p, p)_F$ from crisp input values for variables x_j $(j = 1, \ldots, k)$ is defined as follows,

$$Y = A_1x_1 + A_2x_2 + \cdots + A_kx_k, \tag{5}$$

with symmetric fuzzy coefficients $A_j = (a_j; r_j)_F$ $(j = 1, \ldots, k)$. In this formula, the value of Y is obtained by calculating $(\sum_{j=1}^k a_jc_j, \sum_{j=1}^k r_j|c_j|)$, once explicit values $c_1, \ldots, c_k$ for each given variable. When we have a data set of n number of data, $\{(Y_i; x_{i1}, \cdots, x_{ik})\}_{i=1,\ldots,n}$ with crisp x_{ij} and symmetric fuzzy numbers $Y_i = (y_i; p_i)_F$, we consider the upper regression model and the lower regression model.

For the upper regression model, try to find fuzzy coefficients $A^*_j = (a^*_j; r^*_j)_F$ such that

$$\begin{array}{l} \text{Minimizing: } J(\vec{r}^*) = \sum_{j=1}^k r^*_j \left(\sum_{i=1}^n |x_{ij}| \right), \\ \text{under the condition} \\ Y_i \subseteq Y^*_i = A^*_1x_{i1} + \cdots + A^*_kx_{ik} \qquad (i = 1, \ldots, n). \end{array} \tag{6}$$

The inclusion condition above can be expressed by the following equations, because the shapes of fuzzy set are supposed to be similar

$$\begin{cases} y_i - p_i \geq \sum_{j=1}^k a^*_jx_{ij} - \sum_{j=1}^k r^*_j|x_{ij}| \\ y_i + p_i \leq \sum_{j=1}^k a^*_jx_{ij} + \sum_{j=1}^k r^*_j|x_{ij}| \\ r^*_j \geq 0 \end{cases}. \tag{7}$$

For the lower regression model, try to find fuzzy coefficients $A_{j*} = (a_{j*}; r_{j*})_F$ such that

$$\text{Maximizing: } J(\vec{r_*}) = \sum_{j=1}^{k} r_{j*}\left(\sum_{i=1}^{n} |x_{ij}|\right),$$
under the condition
$$Y_i \supseteq Y_{i*} = A_{1*}x_{i1} + \cdots + A_{k*}x_{ik} \qquad (i = 1, \ldots, n). \tag{8}$$

The inclusion condition above also can be expressed by the following equations.

$$\begin{cases} y_i - p_i \leq \sum_{j=1}^{k} a_{j*}x_{ij} - \sum_{j=1}^{k} r_{j*}|x_{ij}| \\ y_i + p_i \geq \sum_{j=1}^{k} a_{j*}x_{ij} + \sum_{j=1}^{k} r_{j*}|x_{ij}| \\ r_{j*} \geq 0 \end{cases}. \tag{9}$$

For the existence of upper and lower regression model, Guo and Tanaka showed the following theorem.

Theorem 3. *(by Guo and Tanaka in [14])*

1) *There always exists an optimal solution in the upper regression model (6) under (7) .*
2) *There exists an optimal solution in the lower regression model (8) under (9) if and only if there exist $a_{1*}^{(0)}, a_{2*}^{(0)}, \ldots, a_{k*}^{(0)}$ satisfying*

$$y_i - p_i \leq \sum_{j=1}^{k} a_{j*}^{(0)} x_{ij} \leq y_i + p_i \qquad (i = 1, \ldots, n). \tag{10}$$

From the theorem, there might not be any optimal solution for the lower regression model. This problem is caused by the relationship between the number of variables and the number of data. They tried to solve the problems by extending the model into non-linear model which has more formal variables $x_i x_j$ $(i, j = 1, \ldots, k)$ in the following formula.

$$Y = A_0 + \sum_{j=1}^{k} A_j x_j + \sum_{j,l=1}^{k} A_{jl} x_j x_l, \tag{11}$$

with symmetric fuzzy coefficients $A_j,\ A_{jl}$ $(j, l = 1, \ldots, k)$. The right hand side has a quadratic part when considering x_i variables, however, we need to find A_j and A_{jl} for a given data set which minimize or maximize the value, so this might be solved by LP method.

B. Our Linear Model

As a general type of fuzzy number, we consider L-R fuzzy set with monotone decreasing functions satisfying $L(0) = R(0) = 1$ and $L(1) = R(1) = 0$, and denote a L-R fuzzy set by $Y = (y; p, q)_F$, where y is the value giving the maximum uncertainty, e.g., 1, and p, q are left and right range from y, i.e., $y - p$ and $y + q$ give the uncertainty value 0 [3]. We proposed the following type of possibilistic fuzzy regression model

$$Y = A_1 X_1 + A_2 X_2 + \cdots + A_k X_k, \tag{12}$$

with L-R fuzzy variables $Y = (y; p, q)_F$ and $X_j = (x_j; w_j, z_j)_F$ and L-R fuzzy coefficients $A_j = (a_j; r_j, s_j)_F$ $(j = 1, \ldots, k)$.

Let $[Y]_h$ be the support of fuzzy number Y above h-cut line, we have

$$[Y]_h = [y - pL^{-1}(h), y + qR^{-1}(h)],$$
$$[X_j]_h = [x_j - w_j L^{-1}(h), x_j + z_j R^{-1}(h)],$$
$$[A_j]_h = [a_j - r_j L^{-1}(h), a_j + s_j R^{-1}(h)].$$

Applying commonly known multiplication and summation of L-R fuzzy numbers, we have

$$\begin{aligned}[\textstyle\sum_{j=1}^{k} A_j X_j]_h = \ & [\textstyle\sum_{j=1}^{k} (a_j - r_j L^{-1}(h))(x_j - w_j L^{-1}(h)), \\ & \textstyle\sum_{j=1}^{k} (a_j + s_j R^{-1}(h))(x_j + z_j R^{-1}(h))]_h,\end{aligned}$$

and the range of the interval, denoted by J, is calculated by subtracting the left end value from the right end value. Then

$$\begin{aligned} J \ = \ & \textstyle\sum_{j=1}^{k} \{(z_j R^{-1}(h) + w_j L^{-1}(h)) a_j \\ & + (x_j + z_j R^{-1}(h)) R^{-1}(h) s_j \\ & + (x_j - w_j L^{-1}(h)) L^{-1}(h) r_j\}. \end{aligned}$$

Following Guo and Tanaka, we consider upper and lower models, and describe the inclusion relation of the support of Y_i and that of the obtained fuzzy number in the regression model for a given data set.

Now we let ZW_j, XZ_j, XW_j be as follows,

$$\begin{cases} ZW_j = (\sum_{i=1}^{n} z_{ij}) R^{-1}(h) + (\sum_{i=1}^{n} w_{ij}) L^{-1}(h) \\ XZ_j = ((\sum_{i=1}^{n} x_{ij}) + (\sum_{i=1}^{n} z_{ij}) R^{-1}(h)) R^{-1}(h) \\ XW_j = ((\sum_{i=1}^{n} x_{ij}) - (\sum_{i=1}^{n} w_{ij}) L^{-1}(h)) L^{-1}(h) \end{cases}. \tag{13}$$

Then our upper model Y^* is constructed with $A_j^* = (a_j^*; r_j^*, s_j^*)_F$, such that

$$\begin{aligned} &\text{Minimizing: } J(\mathbb{A}^*) = \textstyle\sum_{j=1}^{k} (ZW_j a_j^* + XZ_j s_j^* + XW_j r_j^*), \\ &\text{where } \mathbb{A}^* = (A_1^*, \ldots, A_k^*), \end{aligned} \tag{14}$$

under the condition that for all i

$$\begin{cases} y_i - p_i L^{-1}(h) \geq \sum_{j=1}^{k} (a_j^* - r_j^* L^{-1}(h)) \times \\ \qquad\qquad (x_{ij} - w_{ij} L^{-1}(h)) \\ y_i + q_i R^{-1}(h) \leq \sum_{j=1}^{k} (a_j^* + s_j^* R^{-1}(h)) \times \\ \qquad\qquad (x_{ij} + z_{ij} R^{-1}(h)) \\ r_j^*,\ s_j^* \geq 0 \end{cases}. \tag{15}$$

The lower model Y_* is similarly constructed with $A_{j*} = (a_{j*}; r_{j*}, s_{j*})_F$, such that

$$\begin{aligned} &\text{Maximizing: } J(\mathbb{A}_*) = \textstyle\sum_{j=1}^{k} (ZW_j a_{j*} + XZ_j s_{j*} + XW_j r_{j*}), \\ &\text{where } \mathbb{A}_* = (A_{1*}, \ldots, A_{k*}), \end{aligned} \tag{16}$$

under the condition that for all i

$$\begin{cases} y_i - p_i L^{-1}(h) \leq \sum_{j=1}^{k} (a_{j*} - r_{j*} L^{-1}(h)) \times \\ \qquad\qquad (x_{ij} - w_{ij} L^{-1}(h)) \\ y_i + q_i R^{-1}(h) \geq \sum_{j=1}^{k} (a_{j*} + s_{j*} R^{-1}(h)) \times \\ \qquad\qquad (x_{ij} + z_{ij} R^{-1}(h)) \\ r_{j*},\ s_{j*} \geq 0 \end{cases}. \tag{17}$$

We could also show the following theorem similar to the Theorem 3 on the existence of models.

Theorem 4. *When $x_{ij} - w_{ij} L^{-1}(h) > 0$ $(i = 1, \ldots, n,\ j = 1, \ldots, k)$, then*

1) *There always exists an optimal solution in the upper regression model (14) under (15) .*

2) *There exists an optimal solution in the lower regression model (16) under (17) if and only if there exist* $a^{(0)}_{1*}, a^{(0)}_{2*}, \ldots, a^{(0)}_{k*}$ *satisfying*

$$\begin{cases} y_i - p_i L^{-1}(h) \le \sum_{j=1}^{k}(x_{ij} - w_{ij}L^{-1}(h))a^{(0)}_{j*} \\ y_i + q_i R^{-1}(h) \ge \sum_{j=1}^{k}(x_{ij} + z_{ij}R^{-1}(h))a^{(0)}_{j*} \end{cases} . \quad (18)$$

Proof.

1) If $x_{ij} - w_{ij}L^{-1}(h) \ge 0$ in (15), then $x_{ij} > 0$ from $w_{ij} \ge 0$ and $0 \le L^{-1}(h) \le 1$. Therefore $x_{ij} + z_{ij}R^{-1}(h)$ are also non-negative, and sufficiently large r^*_j and s^*_j satisfy the condition.
2) If there exist $A_{j*} = (a_{j*}; r_{j*}, s_{j*})_F$ $(j = 1, \ldots, k)$ satisfying (17), then we have the condition (18). Conversely, for $a^{(0)}_{j*}$ satisfying (18), put $A^{(0)}_{j*} = (a^{(0)}_{j*}; 0, 0)_F$ and they satisfy the condition (17).

□

Remark1: When the data for independent variables are given in linguistic values, they are usually transformed into fuzzy numbers satisfying the condition $x_{ij} - w_{ij}L^{-1}(h) > 0$ $(i = 1, \ldots, n,\ j = 1, \ldots, k)$. So, the assumptions in the Theorem 4 are not special condition.

Remark2: The condition (18) means the inclusion relation between Y_i and the resulted fuzzy number Y_{i*} of areas between h-cut horizontal line and the base-line ($h = 0$) of them.

Remark2.1: In case of $h = 1$, $L^{-1}(1) = R^{-1}(1) = 0$ and (18) is reduced to

$$y_i = \sum_{j=1}^{k} x_{ij} a^{(0)}_{j*},$$

which means that the line segment of Y_{i*} is in the area of Y_i.

Remark2.2: In case of $h = 0$, $L^{-1}(0) = R^{-1}(0) = 1$ and (18) is reduced to

$$\begin{cases} y_i - p_i \le \sum_{j=1}^{k}(x_{ij} - w_{ij})a^{(0)}_{j*} \le \sum_{j=1}^{k} x_{ij}a^{(0)}_{j*} \\ y_i + q_i \ge \sum_{j=1}^{k}(x_{ij} + z_{ij})a^{(0)}_{j*} \ge \sum_{j=1}^{k} x_{ij}a^{(0)}_{j*} \end{cases} .$$

which means that $Y_{i*} \cap Y_i \ne \phi$.

IV. Regression Method with Kernel

We extend our linear model to a regression model with a kernel-like function, we call modified kernel, on a set of L-R fuzzy number. First we describe a general formula, then give more precise formula as an extension of the polynomial kernel, $k_P(x, y)$, for the case of degree $d = 2$ and the number of explanatory variables $k = 3$ as described in Section II.B.

A. General Formula

We suppose that there exists a function $K(X, Y)$ satisfying only $K(Y, X) = K(X, Y)$ on the product of a set of fuzzy numbers, $\mathcal{X}^k_F \times \mathcal{X}^k_F$ to $\mathcal{X}_F$. Actually, we use a function induced from one of kernels explained in Section II.A if it can be well-defined on fuzzy numbers.

For a given data set of L-R fuzzy numbers, $\{(Y_i, \mathbb{X}_i); i = 1, \ldots, M\}$, where $Y_i = (y_i; p_i, q_i)_F$, $\mathbb{X}_i = (X_{i1}, \ldots X_{ik})_F$ with $X_{ij} = (x_{ij}; w_{ij}, z_{ij})_F$ $(i = 1, \ldots, M, j = 1, \ldots, k)$. We just modify the formula (12) by replacing X_j with $K(\mathbb{X}, \mathbb{X}_i)$, and consider the model

$$Y = A_1K(\mathbb{X}, \mathbb{X}_1) + A_2K(\mathbb{X}, \mathbb{X}_2) + \cdots + A_MK(\mathbb{X}, \mathbb{X}_M), \quad (19)$$

where $\mathbb{X} = (X_1, \ldots, X_k)$ is vector expression of the explanation fuzzy variable and Y is the objective fuzzy variable. For this formula, we can apply our proposed method for the dual model with h-cut. Since the number of data, M, is usually much greater than the number of explanatory variables, k, the possibility of existence for the lower model increases from the Theorem 4.

On the other hand, when M is very big, there will be too many possible fuzzy number coefficients $\{A_i\}$ for both upper and lower model. Thus, try to find smaller set of representer if possible, and denote their number by N. Then fuzzy coefficients $\mathbb{A}^* = (A^*_1, \ldots, A^*_N)$ and $\mathbb{A}_* = (A_{1*}, \ldots, A_{N*})$ are calculated for upper and lower models from the following formulas of fuzzy numbers, respectively,

$$A_1K(\mathbb{X}_i, \tilde{\mathbb{X}}_1) + A_2K(\mathbb{X}_i, \tilde{\mathbb{X}}_2) + \cdots + A_NK(\mathbb{X}_i, \tilde{\mathbb{X}}_N), \quad (20)$$

where $i = 1, \ldots, M$, and $\{\tilde{\mathbb{X}}_l; l = 1, \ldots, N\}$ is a representer.

B. Case of Modified Polynomial Kernel

Here we consider a modified kernel induced from polynomial kernel, $k_P(x, y)$, denoted by $K_F(\mathbb{X}, \tilde{\mathbb{X}}) = (\mathbb{X}^t\tilde{\mathbb{X}} + C)^d$. When we could find $N(={}_{k+d}C_d)$ number of proper value vectors $\vec{\tilde{x}}_l = (\tilde{x}_{l1}, \ldots, \tilde{x}_{lk})$ $(l = 1, \ldots, N)$ for the dense subspace of $\mathcal{H}_{k_P}$, put $\tilde{\mathbb{X}}_l = (\tilde{X}_{l1}, \ldots, \tilde{X}_{lk})$ with $\tilde{X}_{li} = (\tilde{x}_{li}; 0, 0)_F$ $(l = 1, \ldots, N)$.

Now calculate the h-cut of the equation (20) for $C = (c; 0, 0)_F$in the way described in Section III.B. When putting $\vec{x}_i = (x_{i1}, \ldots, x_{ik})$, $\vec{w}_i = (w_{i1}, \ldots, w_{ik})$, $\vec{z}_i = (z_{i1}, \ldots, z_{ik})$, $i = 1, \ldots, M$,we have

$$[\mathbb{X}_i]_h = ([X_{i1}]_h, \ldots, [X_{ik}]_h) = [\vec{x}_i - L^{-1}(h)\vec{w}_i, \vec{x}_i + R^{-1}(h)\vec{z}_i],$$

and the h-cut of the modified kernel is as follows,

$$\begin{aligned} [K(\mathbb{X}_i, \tilde{\mathbb{X}}_l)]_h &= \left(\sum_j^k [X_{ij}]_h[\tilde{X}_{lj}]_h + [C]_h\right)^d \\ &= \left[\left(\sum_{j=1}^k (x_{ij} - w_{ij}L^{-1}(h))\tilde{x}_{lj} + c\right)^d, \right. \\ &\qquad \left. \left(\sum_{j=1}^k (x_{ij} + z_{ij}R^{-1}(h))\tilde{x}_{lj} + c\right)^d\right] \\ &= \left[k_P(\vec{x}_i - L^{-1}(h)\vec{w}_i, \vec{\tilde{x}}_l), k_P(\vec{x}_i + R^{-1}(h)\vec{z}_i, \vec{\tilde{x}}_l)\right]. \end{aligned}$$

Thus we have

$$\begin{aligned} &\left[\sum_{l=1}^N A_lK(\mathbb{X}_i, \tilde{\mathbb{X}}_l)\right]_h \\ &= \left[\sum_{l=1}^N (a_l - r_lL^{-1}(h))k_P(\vec{x}_i - L^{-1}(h)\vec{w}_i, \vec{\tilde{x}}_l), \right. \quad (21) \\ &\qquad \left. \sum_{l=1}^N (a_l + s_lR^{-1}(h))k_P(\vec{x}_i + R^{-1}(h)\vec{z}_i, \vec{\tilde{x}}_l)\right], \end{aligned}$$

and minimizing or maximizing objective value is

$$J(\mathbb{A}) = \sum_{l=1}^{N} a_l \left(\frac{1}{M} \sum_{i=1}^{M} \left(k_P(\vec{x}_i + R^{-1}(h)\vec{z}_i, \vec{\tilde{x}}_l) - k_P(\vec{x}_i - L^{-1}(h)\vec{w}_i, \vec{\tilde{x}}_l) \right) \right) + R^{-1}(h) \sum_{l=1}^{N} s_l \frac{1}{M} \sum_{i=1}^{M} k_P(\vec{x}_i + R^{-1}(h)\vec{z}_i, \vec{\tilde{x}}_l) + L^{-1}(h) \sum_{l=1}^{N} r_l \frac{1}{M} \sum_{i=1}^{M} k_P(\vec{x}_i - L^{-1}(h)\vec{w}_i, \vec{\tilde{x}}_l), \tag{22}$$

where $\vec{\tilde{x}}_l = (\tilde{x}_{l1}, \ldots, \tilde{x}_{lk})$ for $l = 1, \ldots, N$.

Then our upper model Y^* is constructed with $A_j^* = (a_j^*; r_j^*, s_j^*)_F$ minimizing $J(\mathbb{A}^*)$ under the condition that for all $i = 1, \ldots, M$,

$$\begin{cases} y_i - p_i L^{-1}(h) \geq \sum_{l=1}^{N} (a_l^* - r_l^* L^{-1}(h)) \times k_P(\vec{x}_i - L^{-1}(h)\vec{w}_i, \vec{\tilde{x}}_l) \\ y_i + q_i R^{-1}(h) \leq \sum_{l=1}^{N} (a_l^* + s_l^* R^{-1}(h)) \times k_P(\vec{x}_i + R^{-1}(h)\vec{z}_i, \vec{\tilde{x}}_l) \\ r_j^*, \ s_j^* \geq 0 \end{cases}. \tag{23}$$

The lower model Y_* is similarly constructed with $A_{j*} = (a_{j*}; r_{j*}, s_{j*})_F$ maximizing $J(\mathbb{A}_*)$ under the condition that for all $i = 1, \ldots, M$,

$$\begin{cases} y_i - p_i L^{-1}(h) \leq \sum_{l=1}^{N} (a_{l*} - r_{l*} L^{-1}(h)) \times k_P(\vec{x}_i - L^{-1}(h)\vec{w}_i, \vec{\tilde{x}}_l) \\ y_i + q_i R^{-1}(h) \geq \sum_{l=1}^{N} (a_{l*} + s_* l R^{-1}(h)) \times k_P(\vec{x}_i + R^{-1}(h)\vec{z}_i, \vec{\tilde{x}}_l) \\ r_{j*}, \ s_{j*} \geq 0 \end{cases}. \tag{24}$$

We also have the same kind of theorem as Theorem 4.

Theorem 5. *When $k_P(\vec{x}_i - L^{-1}(h)\vec{w}_i, \vec{\tilde{x}}_l) > 0$ and $k_P(\vec{x}_i + R^{-1}(h)\vec{z}_i, \vec{\tilde{x}}_l) > 0$ ($i = 1, \ldots, M$, $l = 1, \ldots, N$), then*

1) *There always exists an optimal solution in the upper regression model under (23) .*
2) *There exists an optimal solution in the lower regression model under (24) if and only if there exist $a_{1*}^{(0)}, \ldots, a_{N*}^{(0)}$ satisfying*

$$\begin{cases} y_i - p_i L^{-1}(h) \leq \sum_{l=1}^{N} k_P(\vec{x}_i - L^{-1}(h)\vec{w}_i, \vec{\tilde{x}}_l) a_{l*}^{(0)} \\ y_i + q_i R^{-1}(h) \geq \sum_{l=1}^{N} k_P(\vec{x}_i + R^{-1}(h)\vec{z}_i, \vec{\tilde{x}}_l) a_{l*}^{(0)} \end{cases}. \tag{25}$$

C. Illustrative Example

As an illustrative example, we consider a polynomial kernel $k_P(x, y)$ of degree $d = 2$ and the number of explanatory variables $k = 3$ cases, so the number of basis for the dense subspace $\mathcal{H}'_k$ of $\mathcal{H}_k$ is $N = 10$. Only considering triangular type fuzzy numbers, i. e., $L = R$ is the linear function from $(0, 1)$ to $(1, 0)$ and $L^{-1}(h) = R^{-1}(h) = 1 - h$, and using the base vectors given in Section II.B, we have

$$\begin{aligned} &\tilde{\mathbb{X}}_l = (\tilde{X}_{l1}, \tilde{X}_{l2}, \tilde{X}_{l3}) \quad (l = 1, \ldots, 10) \text{ with} \\ &\tilde{X}_{11} = (1; 0, 0)_F, \tilde{X}_{22} = (1; 0, 0)_F, \tilde{X}_{33} = (1; 0, 0)_F, \\ &\tilde{X}_{41} = (-1; 0, 0)_F, \tilde{X}_{52} = (-1; 0, 0)_F, \tilde{X}_{63} = (-1; 0, 0)_F, \\ &\tilde{X}_{71} = (1; 0, 0)_F, \tilde{X}_{72} = (1; 0, 0)_F, \\ &\tilde{X}_{82} = (1; 0, 0)_F, \tilde{X}_{83} = (1; 0, 0)_F, \\ &\tilde{X}_{91} = (1; 0, 0)_F, \tilde{X}_{93} = (1; 0, 0)_F, \\ &\tilde{X}_{lj} = (0; , 0, 0)_F \quad \text{otherwise.} \end{aligned}$$

Here, we have $M = 8$ pairs of fuzzy numbers as an example data set shown in Table I. From these fuzzy numbers, calculate $k_P(\vec{x}_i - L^{-1}(h)\vec{w}_i, \vec{\tilde{x}}_l)$ and $k_P(\vec{x}_i + R^{-1}(h)\vec{z}_i, \vec{\tilde{x}}_l)$ for each pair of (i, l) $(i = 1, \ldots, 8, \ l = 1, \ldots, 10)$, then take averages through i for each l. Notice that the calculation is done using $\vec{\tilde{x}}_l$ not $\tilde{X}_{l,i}$.

Next, after setting the constant value for c and the value for h-cut, solve two LP problems, one is for upper model with $\mathbb{A}^*$ and the other is lower model with $\mathbb{A}_*$, satisfying the conditions (23) and (24), respectively.

TABLE I. DATA SET FOR THE ILLUSTRATIVE EXAMPLE

$(y; p, q)_F$	$(x_1; w_1, z_1)_F$	$(x_2; w_2, z_2)_F$	$(x_3; w_3, z_3)_F$
$(3.5; 1.5, 1.5)$	$(1.0; 0.5, 0.1)$	$(2.0; 0.5, 0.5)$	$(3.0; 0.5, 1.0)$
$(4.5; 2.0, 2.0)$	$(2.0; 0.5, 0.1)$	$(2.0; 0.5, 1.0)$	$(3.5; 0.75, 1.0)$
$(7.0; 2.5, 2.5)$	$(3.0; 0.1, 0.0)$	$(6.5; 0.5, 1.5)$	$(5.5; 1.0, 1.25)$
$(9.5; 2.0, 2.0)$	$(2.0; 0.5, 0.1)$	$(9.5; 1.0, 0.5)$	$(10.0; 2.0, 2.5)$
$(11.0; 3.0, 3.0)$	$(4.0; 0.5, 1.0)$	$(9.0; 1.0, 1.0)$	$(10.5; 3.0, 2.5)$
$(6.0; 2.0, 2.0)$	$(2.0; 0.0, 0.0)$	$(3.0; 1.0, 2.0)$	$(2.0; 0.5, 1.0)$
$(8.0; 2.5, 2.5)$	$(3.0; 0.1, 0.0)$	$(5.0; 1.5, 1.5)$	$(5.0; 1.5, 2.0)$
$(9.0; 3.0, 3.0)$	$(3.5; 0.5, 0.0)$	$(4.0; 0.5, 0.5)$	$(6.0; 2.0, 1.25)$

By applying the solver function in MS-EXCEL, when setting $c = 1$ and $h = 0.3$, for the upper model we have

$$A_1^* = (0.218; 0, 0.038)_F, A_6^* = (0.030; 0, 0)_F,$$
$$A_{10}^* = (1.455; 0, 5.230)_F, A_l^* = (0; 0, 0)_F \quad \text{(for other } l\text{)},$$

and

$$Y = A_1^* K(\mathbb{X}, \tilde{\mathbb{X}}_1) + A_6^* K(\mathbb{X}, \tilde{\mathbb{X}}_6) + A_{10}^* K(\mathbb{X}, \tilde{\mathbb{X}}_{10}). \tag{26}$$

For the lower model, we have

$$A_{1*} = (0.160; 0, 0.038)_F, A_{2*} = (0.037; 0, 0)_F,$$
$$A_{3*} = (0.002; 0, 0)_F, A_{10*} = (3.301; 0, 0.167)_F,$$
$$A_{l*} = (0; 0, 0)_F \quad \text{(for other } l\text{)},$$

and

$$Y = A_{1*} K(\mathbb{X}, \tilde{\mathbb{X}}_1) + A_{2*} K(\mathbb{X}, \tilde{\mathbb{X}}_2) + A_{3*} K(\mathbb{X}, \tilde{\mathbb{X}}_3) + A_{10*} K(\mathbb{X}, \tilde{\mathbb{X}}_{10}). \tag{27}$$

Table II describes the correspondence of original values and the resulted values by lower model (27) and by upper model (26). The expression of fuzzy numbers here is not the same as used so far in this paper. These values express the left edge, the center point, and the right edge of each triangular shape. We can see three corresponding fuzzy numbers have no inclusion relation, because they are full numbers before operating h-cut procedure. When looking at the support interval of h-cut of each fuzzy set, we have the set relationship $[Y_*]_h \subset [Y]_h \subset [Y^*]_h$. Figure 1 illustrates the relationship among three fuzzy numbers from the second row in Table II.

We also tried other type of kernels for these test data, and may have some discussion on the fitness.

TABLE II. COMPARISON: Y, Y^*, AND Y_*

$(y-p, y, y+q)$	(y^*-p^*, y^*, y^*+q^*)	(y_*-p_*, y_*, y_*+q_2)
$(2.0, 3.5, 5.0)$	$(2.0, 2.4, 8.1)$	$(3.9, 4.3, 4.8)$
$(2.5, 4.5, 6.5)$	$(2.9, 3.6, 9.5)$	$(4.6, 5.1, 6.0)$
$(4.5, 7.0, 9.5)$	$(5.1, 5.6, 11.8)$	$(7.6, 8.0, 9.8)$
$(7.5, 9.5, 11.5)$	$(4.3, 5.8, 13.1)$	$(7.8, 9.1, 10.2)$
$(8.0, 11.0, 14.0)$	$(7.1, 9.6, 20.2)$	$(9.7, 11.3, 15.5)$
$(4.0, 6.0, 8.0)$	$(3.4, 3.4, 9.1)$	$(5.1, 5.4, 6.6)$
$(5.5, 8.0, 10.5)$	$(5.0, 5.4, 11.9)$	$(6.5, 7.3, 8.8)$
$(6.0, 9.0, 12.0)$	$(5.2, 6.6, 13.0)$	$(6.7, 7.6, 8.7)$

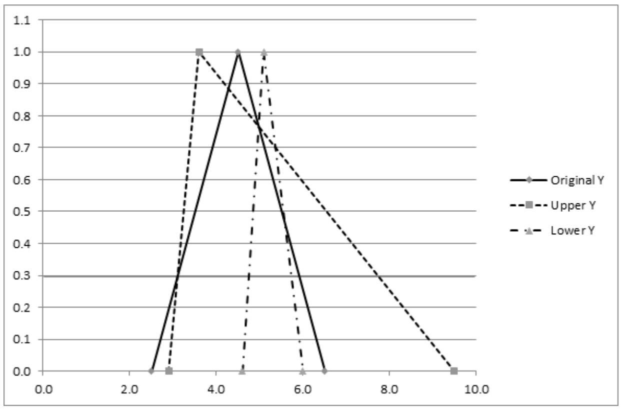


Figure 1. Relationship of Three Fuzzy Numbers

D. Applying to Real Data

The second author of this paper conducted a questionnaire survey on the purchasing behavior of electric assisted bicycle in Japan. The subjects were 102 males and 68 females aged between 20 to 22 years old, and the questionnaire item is a total of 15 items, 6 items are classified as hard function such as selling prices, battery capacity, assist-able maximum distance, charging time, weight, and running cost, and 9 items are classified as soft function such as ride comfortability, ease of operation, etc. Four items of hard function other than the selling price are extracted by using the NGT (Nominal Group Techniques) as the explanatory variables for the price.

According to the questionnaire result, the most important factor for purchasing decision making is the selling price, and the importance degree of the running cost is quite low. Thus we apply our fuzzy regression model to the selling price prediction analysis with four explanatory variables of battery capacity, assist-able maximum distance, charging time, weight. Among 170 responses, the number of valid responses that answered appropriately to these items is 31, and we modified data to obtain the set of the isosceles triangular type fuzzy data since we consider the respondents would have given the values for each of items with uncertainty.

1) Settings and Some Remarks : Here we consider the polynomial kernel $k_P(x, y)$ of degree $d = 2$ and 3 and the number of explanatory variables $k = 4$, so the number of basis for the dense subspace $\mathcal{H}'_k$ of $\mathcal{H}_k$ is $N = 15$ and 35, respectively. Like as in the illustrative example, $L = R$ is the linear function from $(0, 1)$ to $(1, 0)$ and $L^{-1}(h) = R^{-1}(h) = 1-h$. Using the base vectors given in Section II.B.2, we have basis described as follows.

For $d = 2$,

$$\tilde{\mathbb{X}}_l = (\tilde{X}_{l1}, \tilde{X}_{l2}, \tilde{X}_{l3}, \tilde{X}_{l4}) \quad (l = 1, \ldots, 15) \text{ with}$$
$$\tilde{X}_{11} = (1; 0, 0)_F, \tilde{X}_{22} = (1; 0, 0)_F, \tilde{X}_{33} = (1; 0, 0)_F,$$
$$\tilde{X}_{44} = (1; 0, 0)_F, \tilde{X}_{51} = (-1; 0, 0)_F, \tilde{X}_{62} = (-1; 0, 0)_F,$$
$$\tilde{X}_{73} = (-1; 0, 0)_F, \tilde{X}_{84} = (-1; 0, 0)_F,$$
$$\tilde{X}_{91} = (1; 0, 0)_F, \tilde{X}_{92} = (1; 0, 0)_F,$$
$$\tilde{X}_{10\,1} = (1; 0, 0)_F, \tilde{X}_{10\,3} = (1; 0, 0)_F,$$
$$\tilde{X}_{11\,1} = (1; 0, 0)_F, \tilde{X}_{11\,4} = (1; 0, 0)_F,$$
$$\tilde{X}_{12\,2} = (1; 0, 0)_F, \tilde{X}_{12\,3} = (1; 0, 0)_F,$$
$$\tilde{X}_{13\,2} = (1; 0, 0)_F, \tilde{X}_{13\,4} = (1; 0, 0)_F,$$
$$\tilde{X}_{14\,3} = (1; 0, 0)_F, \tilde{X}_{14\,4} = (1; 0, 0)_F,$$
$$\tilde{X}_{lj} = (0; , 0, 0)_F \quad \text{otherwise.}$$

For $d = 3$,

$$\tilde{\mathbb{X}}_l = (\tilde{X}_{l1}, \tilde{X}_{l2}, \tilde{X}_{l3}, \tilde{X}_{l4}) \quad (l = 1, \ldots, 35) \text{ with}$$
$$\tilde{X}_{11} = (1; 0, 0)_F, \tilde{X}_{22} = (1; 0, 0)_F, \tilde{X}_{33} = (1; 0, 0)_F,$$
$$\tilde{X}_{44} = (1; 0, 0)_F, \tilde{X}_{51} = (-1; 0, 0)_F, \tilde{X}_{62} = (-1; 0, 0)_F,$$
$$\tilde{X}_{73} = (-1; 0, 0)_F, \tilde{X}_{84} = (-1; 0, 0)_F, \tilde{X}_{91} = (2; 0, 0)_F,$$
$$\tilde{X}_{10\,2} = (2; 0, 0)_F, \tilde{X}_{11\,3} = (2; 0, 0)_F, \tilde{X}_{12\,4} = (2; 0, 0)_F,$$
$$\tilde{X}_{13\,1} = (1; 0, 0)_F, \tilde{X}_{13\,2} = (1; 0, 0)_F,$$
$$\tilde{X}_{14\,1} = (1; 0, 0)_F, \tilde{X}_{14\,3} = (1; 0, 0)_F,$$
$$\tilde{X}_{15\,1} = (1; 0, 0)_F, \tilde{X}_{15\,4} = (1; 0, 0)_F,$$
$$\tilde{X}_{16\,2} = (1; 0, 0)_F, \tilde{X}_{16\,3} = (1; 0, 0)_F,$$
$$\tilde{X}_{17\,2} = (1; 0, 0)_F, \tilde{X}_{17\,4} = (1; 0, 0)_F,$$
$$\tilde{X}_{18\,3} = (1; 0, 0)_F, \tilde{X}_{18\,4} = (1; 0, 0)_F,$$
$$\tilde{X}_{19\,1} = (-1; 0, 0)_F, \tilde{X}_{19\,2} = (-1; 0, 0)_F,$$
$$\tilde{X}_{20\,1} = (-1; 0, 0)_F, \tilde{X}_{20\,3} = (-1; 0, 0)_F,$$
$$\tilde{X}_{21\,1} = (-1; 0, 0)_F, \tilde{X}_{21\,4} = (-1; 0, 0)_F,$$
$$\tilde{X}_{22\,2} = (-1; 0, 0)_F, \tilde{X}_{22\,3} = (-1; 0, 0)_F,$$
$$\tilde{X}_{23\,2} = (-1; 0, 0)_F, \tilde{X}_{23\,4} = (-1; 0, 0)_F,$$
$$\tilde{X}_{24\,3} = (-1; 0, 0)_F, \tilde{X}_{24\,4} = (-1; 0, 0)_F,$$
$$\tilde{X}_{25\,1} = (1; 0, 0)_F, \tilde{X}_{25\,2} = (-1; 0, 0)_F,$$
$$\tilde{X}_{26\,1} = (1; 0, 0)_F, \tilde{X}_{26\,3} = (-1; 0, 0)_F,$$
$$\tilde{X}_{27\,1} = (1; 0, 0)_F, \tilde{X}_{27\,4} = (-1; 0, 0)_F,$$
$$\tilde{X}_{28\,2} = (1; 0, 0)_F, \tilde{X}_{28\,3} = (-1; 0, 0)_F,$$
$$\tilde{X}_{29\,2} = (1; 0, 0)_F, \tilde{X}_{29\,4} = (-1; 0, 0)_F,$$
$$\tilde{X}_{30\,3} = (1; 0, 0)_F, \tilde{X}_{30\,4} = (-1; 0, 0)_F,$$
$$\tilde{X}_{31\,1} = (1; 0, 0)_F, \tilde{X}_{31\,2} = (1; 0, 0)_F, \tilde{X}_{31\,3} = (1; 0, 0)_F,$$
$$\tilde{X}_{32\,1} = (1; 0, 0)_F, \tilde{X}_{32\,2} = (1; 0, 0)_F, \tilde{X}_{32\,4} = (1; 0, 0)_F,$$
$$\tilde{X}_{33\,1} = (1; 0, 0)_F, \tilde{X}_{33\,3} = (1; 0, 0)_F, \tilde{X}_{33\,4} = (1; 0, 0)_F,$$
$$\tilde{X}_{34\,2} = (1; 0, 0)_F, \tilde{X}_{34\,3} = (1; 0, 0)_F, \tilde{X}_{34\,4} = (1; 0, 0)_F,$$
$$\tilde{X}_{lj} = (0; , 0, 0)_F \quad \text{otherwise.}$$

$M = 31$ pairs of the isosceles triangular type fuzzy numbers are shown in Table III, where $z_i = w_i$ for $i = 1, 2, 3, 4$, and the units of y, $q = p$ is 10,000 yen. From these fuzzy numbers, calculate $k_P(\vec{x}_i - L^{-1}(h)\vec{w}_i, \vec{\tilde{x}}_l)$ and $k_P(\vec{x}_i + R^{-1}(h)\vec{z}_i, \vec{\tilde{x}}_l)$ for each pair of (i, l) $(i = 1, \ldots, 31,\ l = 1, \ldots, 15 \text{ or } 35)$, then take averages through i for each l. Also, notice that the calculation is done using $\vec{\tilde{x}}_l$ not $\tilde{X}_{l,i}$.

Although we preliminary gave the value 1 to the external variable c, for the constant of quadratic polynomial kernel in advance and the solver function in MS-EXCEL is used for the calculation, here the LOOCV (Leave One Out Cross Validation) method is applied to find out proper value for c between a given interval and wrote a program in MATLAB language since MATLAB has a ready-made package for solving LP problems.

In order to judge the properness of each resulted fuzzy number calculated from other 30 data with each value of c during the execution of LOOCV, we adopt the fuzzy similarity measure proposed by Chen [12]. Of course there are various definitions for the fuzzy similarity [11], [7], [9], [25], however, many of them are concerning the trapezoidal type fuzzy numbers and the similar formula is obtained when applying to the triangular type fuzzy numbers.

For two triangular type fuzzy numbers $A = (a_L, a_C, a_R)$ and $B = (b_L, b_C, b_R)$ expressed by left, center, right values in the interval $[0, 1]$, the similarity value of them is defined by

$$Sim(A, B) = 1 - \frac{|a_L - b_L| + 2|a_C - b_C| + |a_R - b_R|}{4}. \tag{28}$$

Since element values in the calculation are not always between 0 and 1, we need to modify them so that they satisfy the condition by transposition and division in the following way.

Put

$$\begin{cases} a'_* = (a_* - min)/d \\ b'_* = (b_* - min)/d \end{cases}, \ for * = R, C, L, \tag{29}$$

where $d = max - min$, $min = min\{a_L, b_L\}$, and $max = max\{a_R, b_R\}$. Then apply the similarity function to the pair of $A' = (a'_L, a'_C, a'_R)$ and $B' = (b'_L, b'_C, b'_R)$.

In the calculation of LOOCV, the program watches the average of $\frac{1}{2}(Sim(Y_i, Y^*_{-i}) + Sim(Y_i, Y_{*-i}))$ $(i = 1, \ldots, 31)$ where Y^*_{-i} (Y_{*-i}) denotes the calculated upper model (resp. lower model) value from the data set given by removing i-th data from the original data set. Then find out the optimal values for the constant c.

TABLE III. DATA SET FOR THE PRICE OF ELECTRIC ASSISTED BICYCLE

$(y;p)_F$	$(x_1;w_1)_F$	$(x_2;w_2)_F$	$(x_3;w_3)_F$	$(x_4;w_4)_F$
(15.0; 5.0)	(7.0; 1.0)	(27.5; 2.5)	(4.0; 1.0)	(15.0; 3.0)
(11.5; 3.5)	(9.0; 3.0)	(35.5; 10.5)	(4.0; 1.0)	(15.0; 5.0)
(8.5; 1.5)	(12.0; 4.0)	(43.0; 7.0)	(26.0; 1.0)	(4.5; 1.5)
(8.5; 1.5)	(11.5; 3.5)	(45.0; 5.0)	(5.0; 1.0)	(30.0; 10.0)
(13.5; 3.5)	(13.5; 1.5)	(45.0; 5.0)	(5.5; 1.5)	(15.0; 3.0)
(11.2.5; 0.25)	(7.5; 1.5)	(47.5; 7.5)	(5.0; 1.0)	(27.5; 2.5)
(11.0; 4.0)	(12.5; 2.5)	(25.0; 5.0)	(5.5; 0.5)	(21.0; 3.0)
(11.5; 3.5)	(11.5; 3.5)	(35.0; 15.0)	(4.0; 1.0)	(10.0; 5.0)
(12.5; 2.5)	(7.0; 1.0)	(27.0; 3.0)	(5.0; 1.0)	(22.5; 2.5)
(11.5; 1.5)	(12.0; 3.0)	(45.0; 5.0)	(3.0; 1.0)	(25.0; 3.0)
(8.75; 1.25)	(11.5; 3.5)	(40.0; 10.0)	(5.0; 1.0)	(27.5; 2.5)
(11.5; 1.5)	(10.5; 1.6)	(45.0; 5.0)	(4.0; 1.0)	(22.5; 2.5)
(9.0; 1.0)	(10.0; 2.0)	(35.0; 5.0)	(11.0; 1.0)	(20.0; 5.0)
(13.5; 1.5)	(12.5; 2.5)	(35.0; 5.0)	(6.5; 0.5)	(27.5; 2.5)
(15.0; 5.0)	(10.0; 5.0)	(40.0; 20.0)	(6.0; 1.0)	(20.0; 5.0)
(8.5; 1.5)	(10.0; 2.0)	(45.0; 5.0)	(5.5; 1.5)	(25.0; 1.0)
(12.5; 2.5)	(7.5; 1.5)	(30.0; 5.0)	(5.0; 1.0)	(25.0; 5.0)
(10.5; 0.5)	(6.5; 1.5)	(27.5; 7.5)	(5.5; 0.5)	(27.5; 2.5)
(11.5; 1.5)	(7.5; 2.5)	(40.0; 10.0)	(9.0; 1.0)	(30.0; 2.0)
(11.0; 1.0)	(10.0; 2.0)	(35.0; 5.0)	(4.5; 1.5)	(19.0; 1.0)
(15.0; 5.0)	(10.0; 2.0)	(35.0; 5.0)	(3.5; 0.5)	(19.0; 1.0)
(12.5; 2.5)	(11.0; 1.0)	(27.5; 7.5)	(7.5; 2.5)	(27.5; 2.5)
(11.0; 1.0)	(7.0; 1.0)	(35.0; 5.0)	(4.0; 1.0)	(25.0; 5.0)
(12.0; 3.0)	(8.0; 2.0)	(32.5; 7.5)	(3.0; 1.0)	(25.0; 5.0)
(11.0; 1.0)	(9.0; 3.0)	(37.5; 7.5)	(5.3; 0.8)	(7.5; 2.5)
(10.0; 2.0)	(10.5; 1.5)	(45.0; 5.0)	(5.5; 0.5)	(17.5; 2.5)
(11.5; 3.5)	(10.0; 2.0)	(40.0; 10.0)	(5.5; 2.5)	(20.0; 5.0)
(9.0; 1.0)	(6.0; 1.0)	(25.0; 5.0)	(5.0; 1.0)	(27.5; 2.5)
(10.75; 1.25)	(11.0; 1.0)	(40.0; 5.0)	(4.5; 0.5)	(25.0; 5.0)
(12.5; 2.5)	(10.0; 2.0)	(35.0; 15.0)	(4.5; 1.5)	(20.5; 4.5)
(12.5; 2.5)	(12.5; 0.5)	(45.0; 5.0)	(11.0; 1.0)	(35.0; 5.0)

2) Results and Discussion: When applying quadratic polynomial kernel, i.e., $d = 2$, we have solutions for lower model in case of $h = 0.3$, 0.4, and the optimal constant obtained by LOOCV is $c = -37$, -32, respectively. Using each optimal constant value, the program re-calculate the fuzzy coefficients $\mathbb{A}^* = (A^*_1, \ldots, A^*_N)$ and $\mathbb{A}_* = (A_{1*}, \ldots, A_{N*})$ for upper and lower models, then the corresponding fuzzy numbers with average similarity values 0.717 and 0.612 are obtained.

Table IV represents the original data of Y, calculated upper model fuzzy numbers Y^* and lower model fuzzy numbers Y_* in case that $h = 0.3$ with the optimal constant $c = -37$. As the notation in the table, $y_R = y - p$, $y_L = y + q$, $y^*_R = y^* - p^*$, $y^*_L = y^* + q^*$, $y_{*R} = y_* - p_*$, $y_{*L} = y_* + q_*$. And, in the "Simil" row, values are calculated by

$$\frac{1}{2}(Sim(Y_i, Y^*_i) + Sim(Y_i, Y_{*i})),$$

for each of data $i = 1, \ldots, 31$.

We can see that all the right side values coincide with the center values both in lower and upper models, and the left side values especially in the upper model is very big. This phenomenon is alleviated in case of $h = 0.4$, however, values for lower model becomes crisp instead.

TABLE IV. COMPARISON: Y, Y^*, AND Y_* BY QUADRATIC POLYNOMIAL KERNEL WITH $h = 0.3$ AND $c = -37$

(y_R, y, y_L)	(y^*_R, y^*, y^*_L)	(y_{*R}, y_*, y_{*L})	Simil
(10, 15, 20)	(8.2, 8.2, 66.5)	(12.4, 12.4, 39.2)	0.75
(8, 11.5, 15)	(8.7, 8.7, 67.1)	(13.4, 13.4, 40.2)	0.74
(7, 8.5, 10)	(9.4, 9.4, 67.7)	(11.4, 11.4, 38.2)	0.72
(7, 8.5, 10)	(5.7, 5.7, 64.0)	(9.4, 9.4, 36.2)	0.74
(10, 13.5, 17)	(9.5, 9.5, 67.8)	(8.7, 8.7, 35.5)	0.74
(11, 11.25, 11.5)	(1.7, 1.7, 60.1)	(10.5, 10.5, 37.3)	0.71
(7, 11, 15)	(9.4, 9.4, 67.8)	(13.9, 13.9, 40.6)	0.74
(8, 11.5, 15)	(6.4, 6.4, 64.7)	(10.8, 10.8, 37.6)	0.75
(10, 12.5, 15)	(8.5, 8.5, 66.8)	(11.6, 11.6, 38.3)	0.75
(10, 11.5, 13)	(10.8, , 10.8, 69.1)	(13.3, 13.3, 40.0)	0.74
(7.5, 8.75, 10)	(8.3, 8.3, 66.7)	(10.9, 10.9, 37.7)	0.73
(10, 11.5, 13)	(8.8, 8.8, 67.2)	(13.3, 13.3, 40.1)	0.73
(8, 9, 10)	(10.2, 10.2, 68.5)	(8.9, 8.9, 35.6)	0.75
(12, 13.5, 15)	(9.7, 9.7, 68.1)	(12.0, 12.0, 38.8)	0.74
(10, 15, 20)	(8.5, 8.5, 66.9)	(11.2, 11.2, 38.0)	0.75
(7, 8.5, 10)	(7.0, 7.0, 65.4)	(11.1, 11.1, 37.8)	0.72
(10, 12.5, 15)	(7.6, 7.6, 66.0)	(11.1, 11.1, 37.9)	0.74
(10, 10.5, 11)	(7.1, 7.1, 65.4)	(10.2, 10.2, 36.9)	0.74
(10, 11.5, 13)	(8.9, 8.9, 67.2)	(10.1, 10.1, 36.9)	0.75
(10, 11, 12)	(9.2, 9.2, 67.5)	(12.6, 12.6, 39.4)	0.73
(10, 15, 20)	(10.3, 10.3, 68.6)	(13.6, 13.6, 40.4)	0.77
(10, 12.5, 15)	(9.6, 9.6, 67.9)	(14.6, 14.6, 41.4)	0.73
(10, 11, 12)	(5.8, 5.8, 64.1)	(10.8, 10.8, 37.6)	0.74
(9, 12, 15)	(7.5, 7.5, 65.9)	(11.3, 11.3, 38.1)	0.75
(10, 11, 12)	(4.0, 4.0, 62.4)	(9.5, 9.5, 36.3)	0.72
(8, 10, 12)	(9.2, 9.2, 67.5)	(11.8, 11.8, 38.5)	0.74
(8, 11.5, 15)	(8.6, 8.6, 67.0)	(11.7, 11.7, 38.5)	0.76
(8, 9, 10)	(7.2, 7.2, 65.6)	(9.6, 9.6, 36.4)	0.74
(9.5, 10.75, 12)	(8.9, 8.9, 67.2)	(11.8, 11.8, 38.6)	0.74
(10, 12.5, 15)	(9.3, 9.3, 67.6)	(12.6, 12.6, 39.3)	0.76
(10, 12.5, 15)	(10.6, 10.6, 68.9)	(9.8, 9.8, 36.6)	0.02

When applying cubic polynomial kernel, i.e., $d = 3$, we have solutions for lower model in case of $h = 0.5, 0.6, 0.7, 0.8, 0.85$ and the optimal constant obtained by LOOCV is $c = 25$, 25, 28, 8, 8, respectively. Using these optimal values, also re-calculate the fuzzy coefficients for upper and lower models, then the corresponding fuzzy numbers with average similarity values 0.733, 0.735, 0.727, 0.720, 0.728 are obtained.

Table V represents the original data of Y, calculated upper model fuzzy numbers Y^* and lower model fuzzy numbers Y_* in case that $h = 0.6$ with the optimal constant $c = 25$.

The center values in both models are very similar to those of the original data, however, left and right values are all big especially the left value of lower model. In case of $h = 0.7$, this phenomenon is somehow alleviated, however, the left value of the upper model becomes bigger. In case of $h = 0.8$, the right side value of upper model and left side value of lower model are very big.

TABLE V. COMPARISON: Y, Y^*, AND Y_* BY CUBIC POLYNOMIAL KERNEL WITH $h = 0.6$ AND $c = 25$

(y_R, y, y_L)	(y^*_R, y^*, y^*_L)	(y_{*R}, y_*, y_{*L})	Simil
$(10, 15, 20)$	$(-22.2, 15.1, 48.4)$	$(-5.8, 14.0, 116.8)$	0.78
$(8, 11.5, 15)$	$(-26.3, 11.1, 44.4)$	$(-7.6, 12.3, 115.0)$	0.77
$(7, 8.5, 10)$	$(-28.3, 9.0, 42.3)$	$(-10.9, 8.9, 111.7)$	0.76
$(7, 8.5, 10)$	$(-29.6, 7.7, 41.0)$	$(-11.8, 8.0, 110.8)$	0.76
$(10, 13.5, 17)$	$(-27.8, 9.6, 42.9)$	$(-7.5, 12.3, 115.1)$	0.76
$(11, 11.25, 11.5)$	$(-28.5, 8.8, 42.1)$	$(-4.9, 15.0, 117.7)$	0.74
$(7, 11, 15)$	$(-24.8, 12.5, 45.8)$	$(-10.5, 9.3, 112.1)$	0.77
$(8, 11.5, 15)$	$(-24.7, 12.7, 45.9)$	$(-8.2, 11.6, 114.4)$	0.77
$(10, 12.5, 15)$	$(-23.4, 14.0, 47.2)$	$(-6.4, 13.5, 116.2)$	0.76
$(10, 11.5, 13)$	$(-26.1, 11.3, 44.5)$	$(-5.8, 14.0, 116.8)$	0.76
$(7.5, 8.75, 10)$	$(-27.2, 10.1, 43.4)$	$(-11.5, 8.4, 111.1)$	0.75
$(10, 11.5, 13)$	$(-29.7, 7.7, 41.0)$	$(-9.1, 10.8, 113.5)$	0.75
$(8, 9, 10)$	$(-29.9, 7.4, 40.7)$	$(-12.5, 7.4, 110.1)$	0.75
$(12, 13.5, 15)$	$(-23.3, 14.0, 47.3)$	$(-7.5, 12.3, 115.1)$	0.76
$(10, 15, 20)$	$(-26.9, 10.4, 43.7)$	$(-12.1, 7.8, 110.5)$	0.76
$(7, 8.5, 10)$	$(-28.7, 8.6, 41.9)$	$(-11.2, 8.6, 111.4)$	0.76
$(10, 12.5, 15)$	$(-24.0, 13.3, 46.6)$	$(-8.5, 11.4, 114.1)$	0.76
$(10, 10.5, 11)$	$(-25.8, 11.6, 44.8)$	$(-10.1, 9.7, 112.5)$	0.75
$(10, 11.5, 13)$	$(-27.9, 9.5, 42.8)$	$(-10.5, 9.3, 112.1)$	0.75
$(10, 11, 12)$	$(-24.2, 13.2, 46.4)$	$(-8.5, 11.4, 114.1)$	0.75
$(10, 15, 20)$	$(-23.1, 14.3, 47.5)$	$(-7.2, 12.7, 115.4)$	0.77
$(10, 12.5, 15)$	$(-23.4, 14.0, 47.2)$	$(-6.8, 13.0, 115.8)$	0.76
$(10, 11, 12)$	$(-25.6, 11.7, 45.0)$	$(-10.2, 9.7, 112.4)$	0.75
$(9, 12, 15)$	$(-24.5, 12.9, 46.2)$	$(-8.6, 11.3, 114.0)$	0.76
$(10, 11, 12)$	$(-28.6, 8.7, 42.0)$	$(-9.6, 10.2, 113.0)$	0.75
$(8, 10, 12)$	$(-30.6, 6.7, 40.0)$	$(-9.6, 10.3, 113.0)$	0.76
$(8, 11.5, 15)$	$(-27.4, 9.9, 43.2)$	$(-11.6, 8.2, 111.0)$	0.76
$(8, 9, 10)$	$(-25.5, 11.9, 45.1)$	$(-10.9, 9.0, 111.7)$	0.75
$(9.5, 10.75, 12)$	$(-26.8, 10.5, 43.8)$	$(-10.7, 9.1, 111.9)$	0.76
$(10, 12.5, 15)$	$(-24.1, 13.3, 46.6)$	$(-8.7, 11.1, 113.9)$	0.76
$(10, 12.5, 15)$	$(-28.8, 8.6, 41.8)$	$(-10.4, 9.4, 112.2)$	0.02

V. CONCLUSION

As an extension of our fuzzy dual linear regression model, we proposed to apply kernel method and give a general formula with a modified kernel of polynomial type. Then, we showed how it works using artificial sample data set for illustration of performance in a simple case.

Although we could see that the kernel method can be incorporated with fuzzy regression model, the effectiveness of our method, depending on data set type, is not yet clear. In the example handling small data, when changing the values slightly, we could not have any solution for the lower model. This infeasibility also occurs by increasing the value of h, which may reduce the degree of freedom of resulted fuzzy number of lower model. Though the number of data is less than the number of base set, the merit of choosing base set is that the number N depends only on the degree of kernel and the number of explanatory variables, and does not depend on the size of data set, M.

In order to construct proper model by applying our model to real data, we need to prepare several types of modified kernel and need to investigate feasibility conditions for the induced LP problem. As we see from the list of calculated model in the last part by applying the method to the set of real data obtained from questionnaire survey on selling price and some factors of electric assisted bicycle, the resulted values contains too many uncertainties and the values of the spread width are monotonous. These phenomenon might be dependent on how to modify the original crisp values to fuzzy values. Therefore we need more raw data expressed in fuzzy values.

In the implementation of LOOCV for determining the external variable, the fuzzy similarity measure is critical. We also need to investigate them as our future works.

REFERENCES

[1] K. Nagata and M. Amagasa, "Decision Making by a Fuzzy Regression Model with Modified Kernel," *Proceedings of The Seventh International Conference on Advances in Information Mining and Management*, IMMM2017, pp. 18-23, 2017.

[2] M. A. Aizerman, E. M. Braverman, and L. I. Rozonoer, "Theoretical foundations of the potential function method in pattern recognition learning," *Automation and Remote Control*, Vol. 25, pp. 821-837, 1964.

[3] A. Alim, F. T. Johora, S. Babu, and A. Sultana, "Elementary Operations on L-R Fuzzy Number," *Advanced in Pure Mathematics*, 5, pp. 131-136, 2015.

[4] M. Amagasa, "Formulation of A Sale Price Prediction Model Based on Fuzzy Regression Analysis," *Operations Research Proceedings 2011*, Springer Berlin Heidelberg, pp. 567-572, 2012.

[5] M. Amagasa and K. Nagata, "Prediction Model with Interval Data - Toward Practical Applications-," *Proceedings of Information Processing and Management of Uncertainty in Knowledge-Based Systems(IPMU 2016)*, Part II, CCIS611, pp. 213-224, 2016.

[6] N. Aronszajn, "Theory of reproducing kernels," *Transaction of the American Mathematical Society*, 68(3), pp.337-404, 1950.

[7] L. Baccour, A. M. Alimi, and R. I. John, " Similarity measures for intuitionistic fuzzy sets: State of the art," *Journal of Intelligent & Fuzzy Systems*, 24, pp. 37-49, 2013.

[8] F. R. Bach and M. I. Jordan, "Kernel independent component analysis," *Journal of Machine Learning Research*, vol. 3, pp. 1-48, 2002.

[9] I. Beg and S. Ashraf, " Similarity Measures for Fuzzy Sets," *Applied and Computational Mathematics*, vol.8 (2), pp. 192-202, 2009.

[10] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," *Proceedings of the fifth annual ACM workshop on Computational learning theory(COLT'92)*, pp. 144-152, ACM Press, 1992.

[11] S-J. Chen, " New Similarity measure of generalized fuzzy numbers based on geometric mean averaging operator," *Proceedings of IEEE International Conference on Fuzzy Systems*, Fuzz-IEEE 2006, Vancouver, Canada, 2006.

[12] S. M. Chen, "New methods for subjective mental workload assessment and fuzzy risk analysis," *Cybernetics and Systems*, vol.27, no.5, pp. 449-472, 1996.

[13] B. T. Gale, *Managing Customer Value*, FREE PRESS, 1994.

[14] P. Guo and H. Tanaka, "Dual Models for Possibilistic Regression Analysis," *Computational Statistics & Data Analysis*, vol.51(1), pp. 252-266, 2006.

[15] Y. Inoue, Y. Uematsu, M. Amagasa, and G. Tomizawa, "The method of setting forecast selling price model by fuzzy regression analysis," *Proceedings of Japan industrial Management Association*, Autumn Congress, pp. 194-195, 1991.

[16] T. Melzer, M. Reiter, and H. Bischof, " Nonlinear feature extraction using generalized canonical correction analysis, " *Proceedings of Internal Conference Artificial Neural Networks(ICANN2001)*, pp. 353-360, 2001.

[17] S. Mika, G. Rätsh, J. Weston, B. Schölkopf, and K.-R. Müller, "Fisher's discriminant analysis with kernels," *Neural Networks for Signal Processing*, vol. 11, pp. 41-48, IEEE, 1999.

[18] R. Rosipal and L. J. Trejo, "Kernel Partial Least Squares Regression in Reproducing Kernel Hilbert Space," *Journal of Machine Learning Research*, 2, pp. 97-123, 2001.

[19] B. Schölkopf, R. Herbrich, and A. J. Smola, " A Generalized Representer Theorem," *Proceedings of 14th Annual Conference on Computational Learning Theory and 5th European Conference on Computational Learning Theory*, pp. 416-426, Springer-Verlag, 2001.

[20] B. Schölkopf, A. J. Smola, and K.-R. Müller, " Nonlinear Component Analysis as a Kernel Eigenvalue Problem," *Neural Computation*, vol. 10, no. 5, pp. 1299-1319, MIT Press, 2006.

[21] J. Shawe-Taylor and N. Cristinanini, "Margin distribution and soft margin, " *Advanced in Large Margin Classifiers*, pp. 349-358, MIT Press, 2000.

[22] H. Tanaka, S. Uejima, and K. Asai, "Linear regression analysis with fuzzy model," IEEE Transactions on Systems, Man and Cybernetics, SMC-12, no.6, pp. 903-907, 1982.

[23] H. Tanaka, "Fuzzy analysis by possibility linear models, " *Fuzzy Sets and Systems*, vol.24, pp. 363-375, 1987.

[24] P. D'Urso, "Linear regression analysis for fuzzy/crisp input and fuzzy/crisp output data," *Computational Statistics & Data Analysis*, 42, pp. 47-72, 2003.

[25] D. Yong, S. Wenkang, D. Feng, and L. Qi, " A new similarity measure of generalized fuzzy numbers and its application to pattern recognition," *Pattern Recognition Letters*, vol.25, pp. 875-883, 2004.

Testing the Feasibility of Residential Wireless Interfaces Virtualization

Antonio da Silva Fariña,
Ana Belén García Hernando
Dept. of Telematic and Electronic Engineering
Universidad Politécnica de Madrid
Madrid, Spain
antonio.dasilva@upm.es,anabelen.garcia@upm.es

Mary Luz Mouronte López

Universidad Francisco de Vitoria
Madrid, Spain
maryluz.mouronte@ufv.es

***Abstract*—This work describes our proposal of leveraging two current networking trends, namely Network Functions Virtualization and Software Defined Networking, to enhance the flexibility with which residential IoT services can be offered to users. Concretely, we advocate for the potential virtualization of all the functionality that lies on top of the in-home wireless communications. These tasks, once virtualized, can be run in the cloud in the form of virtual machines or lightweight containers by using the Network Functions Virtualization capabilities of the Internet Service Provider. This way, the Internet Service Provider would also become a kind of "IoT Service Provider". From a practical perspective, we have chosen Universal Serial Bus as the boundary on top of which the functionality would be virtualized and remotely executed. The main reason for this election is that Universal Serial Bus is widely adopted as a residential interface and there is an ample variety of wireless communications that can be accessed through Universal Serial Bus dongles with the appropriate drivers. Our idea is to tunnel the Universal Serial Bus blocks between the low level IoT gateway at home (which we have called Home Radio Head, or HRH) and the Internet Service Provider in order for the latter to access these flows and treat them accordingly to build IoT services for the residential user. These tunnels could be dynamically provisioned by using Software Defined Networking protocols. In this paper we elaborate not only on our architecture but also on the results we have obtained from tests run on a pilot, considering several networking configurations between the residential environment and the emulated Internet Service Provider premises. These tests range from purely functional proof of concepts to estimations of bandwidth and delays incurred when using our proposal. The figures obtained show that the idea is feasible, and point to cases where executing part of the virtualized functionality inside the Home Radio Head might be necessary for performance or bandwidth usage reasons.**

Keywords-Residential IoT; Network Functions Virtualization; Software Defined Networking; USB interfaces virtualization; USB/IP tunneling.

I. INTRODUCTION

This paper is an extension of a previous one, of the same authors, accepted and presented during the Thirteenth International Conference on Wireless and Mobile Communication (ICWMC 2017) [1]. In that work we introduced our proposal for an architecture that leverages two current networking trends, namely Network Functions Virtualization (NFV) [2][3] and Software Defined Networking (SDN) [4], to augment the flexibility when offering a portfolio of IoT services to residential users. This proposal consists in the virtualization and remote execution of the functionality that lies on top of the residential IoT wireless communications. This way, different IoT manufacturers' hardware can be more easily supported, since the specific drivers and all the upper level tasks can be run by the Services Provider and even activated in a "plug and play" manner for the users.

By residential IoT we mean the part of the Internet of Things that is or will be deployed mainly for private homes or small companies (i.e., SOHO - Small Office Home Office). These environments do not usually pose such strict constraints regarding availability, safety or tolerance to harsh conditions as industrial, environmental, healthcare or outdoor scenarios do (to name a few). In [5], different IoT application domains are compared in terms of traffic characteristics and QoS requirements; in the case of "smart buildings and living", there are small network sizes (tens or at most hundredths of nodes) and short coverage areas, the traffic rate is low (or medium at most), and the tolerance to delay usually ranges between seconds to tenths of seconds.

However, the high capillarity of homes with Internet access that will deploy IoT services, and the ample variety of proprietary solutions on the market, make it very convenient to be capable of offering a unified and virtualized support for all the residential IoT solutions.

Nowadays, application-layer gateways are usually needed to provide connectivity to IoT devices in the home. Current gateways mix network connectivity, in-network processing, and user interface functions. We share the view of [6] by which separating these functions would improve the connectivity potential for IoT devices. In fact, there are different publications that consider NFV and SDN as two elements of the new ISPs (Internet Service Providers) architecture to flexibly support IoT [7].

Our specific approach requires for the low level flows of information to be tunneled and consumed at the ISP side, and we opted for Universal Serial Bus (USB) as the boundary between the purely wireless in-home communication and the raw information that is to be processed remotely. This way, a generic and programmable gateway would be present at home, called Home Radio Head (HRH) (a name inspired on the Remote Radio Heads (RRH) present in some cellular network deployments), which does not need to implement any IoT vendor-specific function above the wireless communications provided by

USB dongles. USB interfaces would be virtualized and managed remotely thanks to the establishment of tunnels between the HRH and the ISP.

At the remote end of these tunnels, a NFV infrastructure hosts the IoT applications and management functionality implemented as a set of Virtual Network Functions (VNFs). The virtualization of USB interfaces allows to reach economies of scale by offering a reasonably inexpensive customer premises equipment supporting a wide variety of home wireless communications.

Moreover, we propose to also introduce the programmability of SDN into our architecture, so that the USB tunnels are dynamically provisioned by an SDN controller that communicates with the HRH through a southbound interface such as OpenFlow [8].

In order to reduce the necessary bandwidth between home and the ISP, or if the delay/latency requirements are very stringent for a specific application, some processing can be carried out inside the customer premises by downloading and running lightweight Virtual Machine (VM) containers [9] in the HRH.

In our paper [1] we also showed our first results on the feasibility of this approach, consisting on estimations of the bandwidth attainable through USB/IP tunneling [10] between the home and the ISP under different networking scenarios.

This paper elaborates more on the feasibility and quantitative study of our proposal. Firstly, an actual low rate video flow generated by a USB WebCam has been successfully transported and visualized by a virtual function located in an emulated ISP. Secondly, round trip time (RTT) measurements are taken and compared for local and remote (virtualized) scenarios. This gives the idea of when it is necessary to execute some of the virtualized functionality inside the HRH, which, besides being SDN manageable, could incorporate some kind of lightweight virtualization capacities as indicated above.

The rest of the paper is organized as follows: Section II summarizes the background and some related work. Section III describes our proposed architecture and its building blocks. Section IV explains the experimental setup we have implemented and discusses the results. Finally, in Section V, we provide the conclusion and our foreseen future work.

II. Background

In this section we firstly review the work related to the use of NFV and SDN in home environments. Then we elaborate on the concept of the virtualization and remote execution of radio functionality, which is used in current cellular networks and inspires our proposal for a HRH. After that, the main lightweight virtualization options that could be useful for executing part of the VNFs in the HRH are described. Finally we summarize the current literature on the virtualization of residential IoT.

A. *NFV and SDN in Home Environments*

NFV leverages commodity storage, networking and processing equipment in order to execute, through the use of a virtualization layer (sometimes called hypervisor), sophisticated network functionality on top of a virtualized infrastructure. It may be used to combine the available resources in a network by dividing the bandwidth into channels or slices, each of which is independent from the others. NFV allows multiple service providers to construct different separate and isolated virtual networks, which share physical resources.

The standardization of NFV started with ETSI, and several use cases have been defined in [11], such as the virtualization of the home environment and the virtualization of Internet of Things.

With SDN, the control plane (in which the logical procedures supporting the networking protocols and the most important decisions are made) is separated from the data plane (in which the forwarding of packets on the most suitable interface towards the intended destination is carried out). SDN is an excellent mechanism to do Traffic Engineering (TE) and exploit effectively the network resources in an IoT scenario.

These two technologies, far from being incompatible, are increasingly being considered together in new architectures. The main use cases of SDN and NFV in the home deal with pure networking tasks, and more specifically with the virtualization of the Customer Premises Equipment (CPE). There are also some recent proposals to augment the scope of cloud computing, NFV and SDN and integrate some IoT (basically sensing and actuation) capabilities into their frameworks [12][13].

In some use cases [14], hardware middleboxes are deployed by cloud providers, executing several network functions, and enhancing the cloud capability. To solve the costs, manageability, and performance overhead problems, NFV has been proposed as a good solution and therefore, software applications have been deployed in place of the hardware middleboxes. The high computing power brought by NFV would provide rich-media functionalities to thin customer devices and would change the way the multimedia services and applications are used.

The paper [6] proposes an architecture that leverages the increasingly ubiquitous presence of Bluetooth Low Energy radios to connect IoT devices to the Internet; several example applications are shown, and several research challenges in its implementation are investigated.

Home environments where the hardware resources are shared efficiently and overviews of different virtualization mechanisms to cooperate among home networks have been investigated, comparing methods and concepts [15]. Key technical challenges behind this idea, such as dynamic allocation, migration, and orchestration of virtual machines across wide areas of interconnected edge networks have been analyzed [16], and new mechanisms to allow that residential users control the access network resources and manage different types of traffic to fulfil the Quality of Service (QoS) requirements have been proposed [17]. Some studies on how to place NFV to satisfy the QoS requirements have also been made [18][19].

The work [20] describes an efficient network management proposal by means of home gateways that focus on supervising the network traffic flows, executing a per-flow management, and implementing a custom DHCP (Dynamic Host Configuration Protocol) to enable traffic segregation and its proper measurement at the IP (Internet Protocol) layer. The authors of [21] discuss novel solutions to build a virtual residential gateway using an SDN controller at the service provider side to manage services in home. Another research on residential gateways that can be controlled and managed remotely via an SDN controller, adjusting and troubleshooting the residential network, can be found in [22].

No doubt, security is an important issue when it comes to controlling and managing lots of in-home residential equipment. The security challenges in SDN/NFV [23] and the feasibility of extending the current NFV orchestrator to provide security mechanisms have been recently studied [24]. The proposed security solutions supervise parameters, generate access control policies, and apply them through the underlying infrastructure. Most solutions can work together with the NFV orchestrator to enable fine-grained access management to safeguard services and resources.

The novelty of our proposal lies in the fact that we virtualize all the functionality above the very low level communications between in-home IoT devices and the HRH. This augments the flexibility and can even lead to a real "plug and play" support of new hardware on the part of the ISP, as long as there is a USB dongle plugged in the HRH with which the sensors and actuators communicate.

Our architecture can be seen as one possible realization of a specific instance of the use case "Virtualization of Internet of Things (IoT)" presented in [11], including as an additional element the use of SDN.

The existing literature on the usage of NFV and/or SDN for home environments is usually dependent on the existence of an in-home hub or concentrator (sometimes called Customer Premises Equipment). Many of these proposals include the virtualization of part or all the functionality of this element, whereas ours is, to the best of our knowledge, the first that advocates for the virtualization of all the vendor-specific IoT functionality above a very low-level interface (USB in our case), moving everything else to the ISP. This substitutes the IoT gateway that is maintained in many proposals by a generic HRH easily reusable for new IoT products.

We stress here that our proposal does not prevent the existence of higher-level functions such as service publication and recognition, data analysis or application-specific processing. These functions can be present above the virtual USB points of presence at the NFV infrastructure on the ISP side, in the form of one or several VNFs.

B. Virtualization and Remote Execution of Radio Functionality

Our HRH is inspired by the Remote Radio Head (RRH) approach used in cellular wireless access networks, which aims to move wireless baseband processing to the cloud. This approach has a high cost in terms of bandwidth that is solved using dedicated high-speed lines connecting the RRH with the Base Band Unit (BBU) at the edge of the core network.

Recent researches exist on RRH, such as [25], which describes the process of moving from RRH scenarios where the BBU are deployed at the base of the cell tower, to an architecture in which the BBU are located in several centralized locations. The authors of [26] present an overview of the state of the art on Cloud Radio Access Network (C-RAN) research, with focus on front-haul compression, baseband processing, medium access management, resource allocation, system-level requirements and standardization works.

We find that the virtualization of higher level functionality is appealing not only for cellular networks but also for residential environments. However, the specificities of local area protocols, inherently different from cellular wireless, make it necessary to assess to what extent and under what circumstances this externalization of functions is feasible. In this paper we provide some results obtained from tests that range from the feasibility of consuming actual traffic generated by a WebCam to the measurement of round trip time delays under different networking scenarios between the home and the ISP.

C. Lightweight Virtualization Environments

The orchestration and maintenance of the software running on the gateways in large-scale deployments is a challenging task. There are studies, such as [27], that evaluate the performance of the container-based approach compared to a hypervisor-based virtualization when running on gateway devices. The comparison between traditional heavy VMs, Unikernels and Docker containers is shown in Fig. 1. Unikernels [28] are specialized single-purpose operating systems, which are several magnitudes smaller than general-purpose OS (Operating Systems); they can also be used in small Internet-of-Things (IoT) devices intended to execute a specific software application. On their side, Docker containers are a type of lightweight VM that can be easily deployed in inexpensive common single board computers like Raspberry Pi. [27][29][30] also conclude that Docker presents better performance than the traditional VM as it has no guest operating system and its overhead is considerably reduced. The lightweight virtualization has been also shown to be a suitable technique for auditing applications, detecting intrusions, and recovering systems from attacks and errors [31].

Ultimately the level of security guaranteed by applications developed within containers has become an important concern [32]. The latest versions of Docker have included several security enhancements to solve several security issues. Even more, Docker team continuously publishes guidelines in order to avoid security threats and build safer Docker ecosystems [32][33]. Additionally, a developer's tool that allows examining many security issues within virtualized applications has been implemented as a collaboration work between Docker and the Center for Internet Security [34].

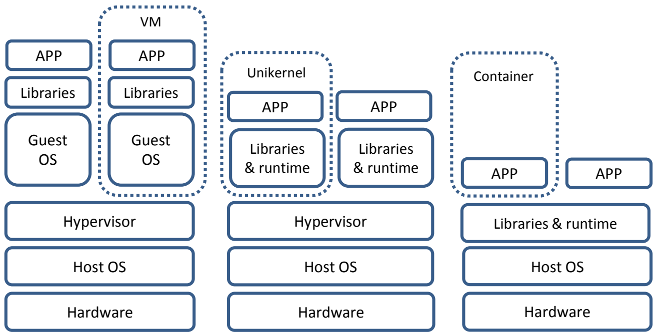


Figure 1. Comparison of traditional VM (left), Unikernels (center) and Container (right) virtualization architectures.

In any of its forms, this lightweight Platform-as-a-Service (PaaS) environment is very useful to deploy custom home sensors preprocessing functionality at runtime. With it, both the necessary bandwidth to the cloud and the round trip delay for those functions can be reduced.

The possibility of distributing the resource-intensive functions in intermediate points between the end devices (e.g., sensors and actuators) and the cloud is aligned with the philosophy behind fog computing approaches. This distribution has to be done transparently for the users, and a certain amount of intelligence is needed to manage a fog scenario, preferably in an open and interoperable manner. IoT and 5G [35] are two of the drivers that currently push the activity in fog computing.

Recent researches [35] describe novel IoT architectures based on SDN and Fog computing, where SDN provides a centralized network control plane, which executes complex mechanisms for traffic and resource management, and fog computing allows that much information is processed and managed at the network edge, being able to support applications with very low latency requirements. Also related to this, in [36] the authors review the distributed approach to NFV, discuss phased NFV deployment and present critical factors to take such functionalities into account at the customer edge.

Our proposal is in a way aligned and compatible with fog/edge computing paradigms, in the sense that we do not restrain the locations at which the IoT specific functionality has to be run. This depends on the NFV infrastructure that the ISP has, and on the policies in place to decide how to split the different VNFs that form a specific service. Functions that are more time-sensitive (e.g., reactions to events that raise alarms) can be executed near (or at) the network edge or even inside the HRH, whereas more delay tolerant tasks (e.g., HVAC -Heating, Ventilation, & Air Conditioning- activation due to temperature changes or statistics collection) can be run in more centralized locations to be shared by more customers.

D. Virtualization of Residential Internet of Things

Building functions that cope with all the diversity that the home IoT products present is not feasible, at least currently and at a reasonable cost, for the residential user. However, for an Internet Service Provider (ISP) this would be much easier, especially if these functions are offered as services to its customers and economies of scale can be applied. The ISP should virtualize the actual physical infrastructure of its customers to deliver a set of general and reusable services [37]. In fact, sensing as a service (S2aaS) architectural proposals are specifically concerned with the organizational relationships between the different components and omit details about short range components communications as well as other technical aspects [38].

This is a very active research field. Among the recent works in this area we highlight the following. In [39], the authors survey the state of the art on the application of SDN and NFV to IoT. They provide a description of the possible implementation aspects for both technologies.

The work [12] highlights some IoT challenges that the network and the IT (Information Technologies) infrastructure will face. The NFV and SDN benefits are presented from the point of view of the network operator. The authors present a new multi-layered IoT architecture involving SDN and NFV, and they show how the proposed architecture is able to cope with the identified IoT challenges.

In [13], the authors discuss the usage of NFV technologies and construct a virtual advanced metering infrastructure (AMI) network to transmit energy-related information in a dependable and cost-effective way. The reliability, availability and cost of the new architecture is analyzed and compared to current AMIs. Another example of a specific application is found in [40], in which the authors propose a Logical Access Point-based Mobility Management (LAPM) scheme for WLAN (Wireless Local Area Network), based on an extended SDN/NFV abstraction, which outsources the IEEE 802.11 protocol stack complexity to a centralized controller.

In [6], an architecture that leverages the increasingly ubiquitous presence of Bluetooth Low Energy radios to connect IoT peripherals to the Internet is proposed. The authors propose the use of mobile devices (i.e., Laptops, Smartphones and tablets) as gateways. The same approach is followed in [41], where the use of smartphones running as gateway bridges with Bluetooth-enabled devices in a home environment is evaluated.

We finally mention the work [42], in which a new user-centric management architecture is proposed, to increase the active engagement of residential users in the management tasks of their own networks, improving the usability of the network and facilitating the provision of new services. The proposed architecture combines the SDN and NFV approaches. Additionally, the user-centricity is achieved by implementing interaction and management layers. These layers together constitute a residential network management application. The interaction layer, which can be deployed over different devices, hosts the application that allows the user to configure the network and receive notifications. The interaction layer interacts with the management layer by means of a REST API (Representational State Transfer Application Programming Interface).

III. Proposed Architecture

In our previous work [37], we described our proposal for an architecture that leverages NFV and SDN to offer a portfolio of IoT services to residential users.

The virtualization of IoT vendor-specific functionality, together with the presence of a cost-effective and generic customer premises equipment called HRH, would bring economies of scale, easier updates and faster support for new IoT products, among others. This in-home HRH:

- Would not need to implement IoT vendor-specific functions, since these functions would be virtualized and run on a standard NFV infrastructure.
- Would send the in-home raw layer 1 physical flows to the ISP by using tunnels.
- Would be SDN-manageable to establish and maintain the aforementioned tunnels in a standardized manner.

With the work described in this paper we elaborate further on this architecture and decide to establish USB as the vendor-agnostic frontier between the in-home radio flows and the vendor-specific functionality that would be implemented as VNFs. In Fig. 2 we show the high-level view of our proposal.

Our architecture is based on the following principles:

- The support of varied short-range wireless interfaces widely used by residential IoT products (e.g., WiFi, Bluetooth, ZWAVE or ZigBee). These radio flows are processed locally at low level and exposed to the HRH as standard USB interfaces.
- The establishment of tunnels between the HRH and the virtualization infrastructure at the ISP side, in order to propagate transparently the USB Request Blocks (URBs) that are to be processed by VNFs. The establishment of these tunnels is programmable by using SDN, i.e., the HRH supports Openflow (or any other SDN-compliant southbound interface).
- The vendor-specific functionality, which is realized by means of one or several VNFs (represented as "vHOME" virtual machines in Fig. 2), can be distributed if necessary. In this line, the HRH supports a lightweight virtualization environment on which a subset of the VNFs necessary for a service can be downloaded and executed, as commanded by the Virtualized Infrastructure Manager (VIM), see Fig. 3.

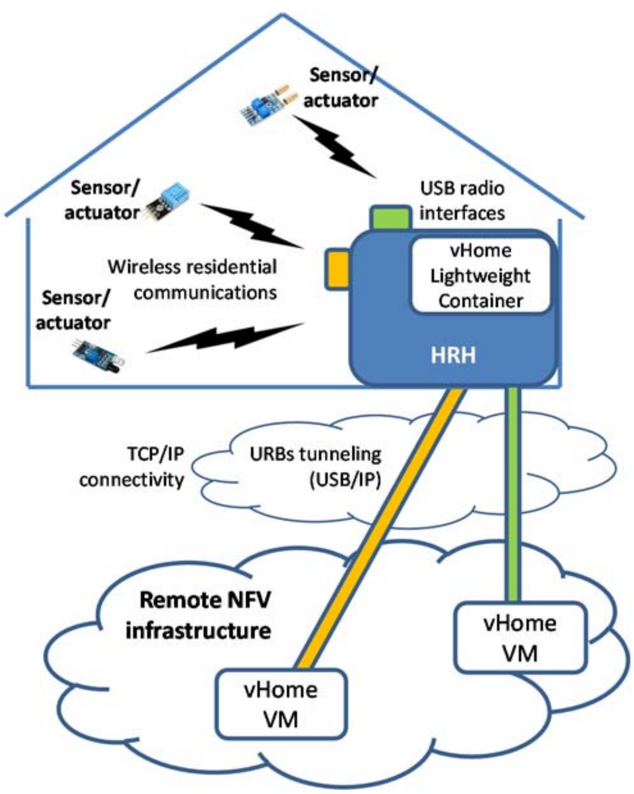


Figure 2. High-level architecture for the remote virtualization of wireless residential communications by means of USB tunneling.

Some of these architectural elements were already drafted in our work [37], such as the use of NFV and SDN, the support of different short-range wireless communications in home, the virtualization of higher-level functionality at the ISP and the possibility of having a lightweight virtualization environment in the HRH. What our current proposal advances with respect to [37] is a much clearer and concrete boundary on what is executed in home and what is virtualized at the ISP, by the selection of USB as the technology whose blocks are going to be tunneled, instead of considering generic "raw layer-1 flows" either at bit or frame level. This way, the "virtual NICs" (virtual Network Interface Cards) that were present in [37] become "vUSBs" in Fig. 3, and the HRH becomes a device with the possibility of having several USB dongles plugged in (to support different wireless or wired IoT products), instead of having to embed the different wireless technologies itself.

Even if establishing USB as the boundary between the in-home and the ISP functionalities can be considered as a limitation to our more general initial architecture, the fact that it is a widely used interface for residential consumer electronics clearly brings advantages with respect to the feasibility of deploying our solution in a wide scale. The fact that the lowest level wireless functions are going to be embedded into the USB dongle is also convenient, since the first establishment of a flow by SDN procedures can take significantly longer than subsequent transmissions, and that initial delay might hinder the beginning of the wireless interactions. Moreover, the HRH becomes a more general device since USB is a well-known technology, and the support of additional wireless technologies would only require to have the adequate USB dongle, acquired together with the rest of the IoT hardware from the vendor. This technological and architectural decision has also allowed us to prove the feasibility of our proposal, as will be explained later in the paper.

In the following sub-sections, we elaborate further on the implications, advantages and rationale behind each of our main architectural design decisions.

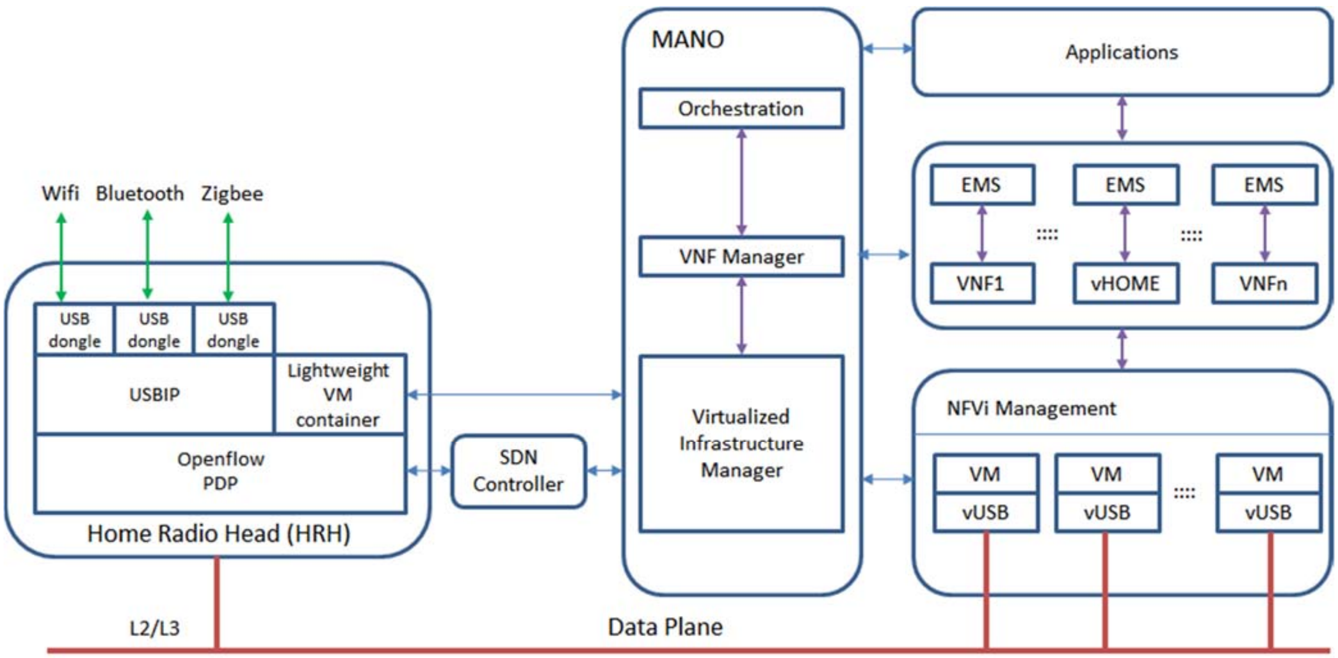


Figure 3. SDN and NFV as considered in our proposal.

A. *Virtualization of USB Interfaces*

USB Redirection consists of plugging an external device into a USB port on a local endpoint and accessing that device from within a remote system or application. The rationale behind the virtualization of USB interfaces that we include in our proposal is threefold. Firstly, USB is widely supported by IoT vendors in residential environments. Secondly, the existence of USB dongles at reasonable prices and easily interchangeable is very convenient for the residential market. Lastly, fast prototyping of proof of concepts becomes possible with general purpose equipment (see Section IV below).

Moreover, virtualizing the functionality above the USB interface would make various existing IoT products immediately available through our proposed schema. New products also supporting USB would become available to customers almost in a "plug-and-play" manner as long as ISPs supported the adequate virtualized drivers.

Each tunnel would correspond to a new IoT product that utilizes a specific wireless technology (see Fig. 2), and the specific drivers and all upper functionality would be placed at the other end of each tunnel, on the ISP side. To implement this idea, we propose to use USB/IP [10], a means of sharing USB devices over a TCP/IP (Transmission Control Protocol / Internet Protocol) network by encapsulating USB messages between a server (the equipment with the USB device physically connected) and a remote client.

B. *SDN-Programmable Data Path*

In order to provide a flexible configuration, the URB flows should be dynamically provisioned and managed. The HRH would benefit from a generic datapath that is programmable by following the SDN principles. The concrete policies to be applied to the establishment of the tunnels would be implemented and enforced by an SDN controller, and a southbound Openflow-programmable datapath has to be supported by the HRH. This way, SDN advantages brought by the software definition of networking configuration are present in our scenario. Also, our HRH would be more easily integrated with an SDN-based residential gateway as proposed in [42].

C. *Lightweight Docker Containers*

Under certain circumstances, it might be convenient or even necessary that a subset of the vendor-specific functionality is run inside the customer premises. This might be the case for complying with stringent delay requirements or for saving uplink/downlink bandwidth. We propose to provide a light virtualization environment, based either on light virtual machines or on Docker containers, inside the HRH, in which specific modules can be downloaded and executed locally when commanded by the NFV management and orchestration layer.

This distribution of functionality has to be done transparently, without the user being aware of the decomposition of the global service into different modules that may be executed at different points.

D. *SDN/NFV Relationship*

Our HRH follows the principles of both SDN and NFV. As such, it contains on one hand a generic and programmable networking datapath, and on the other hand a lightweight virtualization environment. The former offers a standard SDN southbound interface so that the SDN controller can provision the USB tunnels dynamically. The latter is formally part of the virtualized infrastructure that has to be managed by the VIM, as per the NFV architecture.

Fig. 3 is an enhanced version of a figure we included in our previous paper [37]. We have completed the modules and technologies inside HRH, and we have also made the virtualization capabilities of HRH explicit. The SDN controller that is functionally located between the VIM and the HRH (see Fig. 3) can itself be implemented as another VNF, this way leveraging the existing NFV infrastructure.

IV. Experimental Setups and Results

In order to validate our approach, we have carried out several experimental setups. To better assess the reproducibility of these experiments, we highlight here that all the hardware used is inexpensive and off-the-shelf, and all the software is open source. In the same way, and to guarantee the generality of the tests carried out, the test benches are implemented using both Windows and Linux as guest operating systems.

A. *USB Redirection Tests*

We have implemented the experimental setup shown in Fig. 4. To act as HRH, we have equipped a Raspberry Pi 3 with a USB/IP server running on Raspbian OS. This HRH is located inside the Smart Home that the Universidad Politécnica de Madrid has in its South Campus. Both a generic USB WebCam and a USB mass storage device (i.e., a USB pendrive) are connected to the HRH.

The ISP side is emulated by means of a Windows PC equipped with a VirtualBox hypervisor. On top of this virtualization infrastructure, a guest OS is run that contains a USB/IP client. This client is in charge of terminating the USB tunnels and offering the virtualized USB dongles to the guest OS as if they were local. On top of these virtualized USB dongles, the specific functionality can be deployed.

We have designed and executed two USB redirection tests. The first one is actually composed of different measurements. The objective of this composed test is to estimate the bandwidth that would be available through the USB tunneling infrastructure for different local-remote networking scenarios. To perform this estimation, we have run several write and read tests on the regular USB pendrive and have measured the performance of those operations. Four local-remote setups have been considered:

- Local: This is the baseline that gives us the actual write/read capacity of the USB device. Both operations are performed on a pendrive connected to the same node.
- Same network: The HRH and the emulated ISP are connected to the same Ethernet network. In our case, it is a 100BASE-TX wired Ethernet connection.
- Madrid-Alcalá: The HRH is located inside the Smart Home in the South Campus of UPM (Universidad Politécnica de Madrid), whereas the emulated ISP is connected to a residential network in Alcalá de Henares, a village on the outskirts of Madrid. They are around 30 km. apart from each other. The residential network has a 50 MB/s Hybrid fiber-coaxial (HFC) internet connection.
- Madrid-Galicia: The HRH is located inside the Smart Home in the South Campus of UPM, whereas the emulated ISP is connected through a WIFI access point in a public library in Galicia (North of Spain). They are around 500 km. apart from each other. The internet connection of the library, which is provided by a public ISP, is a 100 MB/s HFC access shared by all users.

These setups are chosen to consider scenarios in which not only the distance but also the expected quality of the networking accesses is varied. We have compiled the obtained results in Table I below.

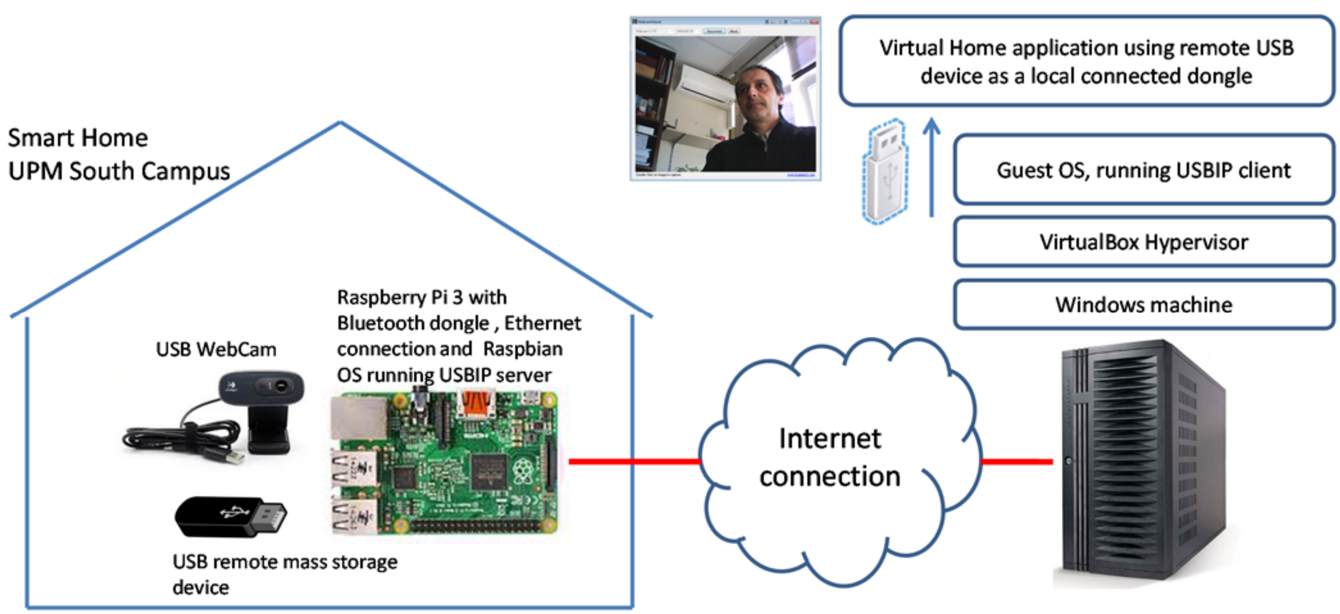


Figure 4. USB redirection tests setup.

TABLE I. WRITE / READ PERFORMANCE UNDER SEVERAL LOCAL-REMOTE CONFIGURATIONS

Configuration	Write	Read
Local	7.34 MB/s	17.5 MB/s
Same network	4.32 MB/s	6 MB/s
Madrid-Alcalá	0.9 MB/s	1 MB/s
Madrid-Galicia	0.4 MB/s	0.5 MB/s

We have specified the concrete type of Internet access that is present at each setup location to give a better idea of the influence that this might have on the final perceived figures included in Table I. Even though we are aware that different specific locations would have thrown different numbers, we consider that the objective of demonstrating the feasibility of virtualizing and executing remotely many of the usual residential functions is reasonably well attained.

Anyway, these bandwidth measures should be taken as a starting point, because they have been obtained in controlled environments that implement per-user traffic shaping or bandwidth limits policies to control peer to peer communications. Nonetheless, even in the most disadvantageous scenario, the bandwidth estimations show that for sporadic or periodical sensor readings (such as temperature or humidity) and for short actuator orders (such as lights on/off control), it is feasible to execute all virtualized functions remotely. In fact, most smart home sensing-and-actuation applications are supposed to present low rates to the network, since each sensor will usually produce one reading (one message) every several minutes [5]. Even in the case of having tenths or near one hundred sensors, it is clear that the traffic rate produced by these applications is compatible with the measurements we have obtained.

Even low-rate video, which would generate a significantly higher throughput than sensor and actuator readings and orders, can be remotely tunneled if necessary: as an example a Youtube video with a resolution of 360p consumes around 0.3/0.4 MB/s. However, in the case of more intensive multimedia traffic, such as a video camera output with higher resolution, it might be necessary to download and execute some of the processing functionality into the HRH, in order to consume less bandwidth towards the ISP. This decision can be made on the basis of bandwidth or delay measurements with each residential subscriber, which could be easily taken by the ISP.

The second USB redirection test evaluates the capability of tunneling periodic USB traffic and processing it in a virtualized environment. To do this, we have used a conventional USB WebCam.

There exist different types of USB transfers, e.g., isochronous transfers are meant for transmitting real-time information such as audio and video, and must be sent at a constant rate. USB isochronous data streams are allocated a dedicated portion of USB bandwidth to ensure that data can be delivered at the desired rate. An isochronous pipe sends a new data packet in every frame, regardless of whether the delivery of the previous packet was successful or not.

On the other hand, interrupt transfers are intended for devices that send and receive data asynchronously. The interrupt transfer type guarantees a maximum service period and that delivery will be re-attempted in the next period if an error occurs on the bus. This transfer protocol is ideal for time sensitive applications because it has a guaranteed bound latency. A typical example is a USB mouse peripheral.

To analyze closely the local USB flow generated by the WebCam we have used Wireshark. Wireshark is a widely used network protocol sniffer, which can also be used to capture raw USB traffic on local endpoints. Fig. 5 shows the result of capturing the traffic in the local WebCam USB connection. As can be seen, in this case the device uses an interrupt transfer mode (even if the isochronous type may seem more adequate). The bandwidth required by the WebCam is not high since the data packets are seven bytes long.

Once this is known, a purely functional test was made. The USB WebCam was plugged in a local endpoint of the raspberry board and the traffic was transmitted using USB/IP to the ISP side. This way, the device was made available to the VM running in the cloud (in the emulated ISP). An off-the-shelf open source WebCam Viewer was used to open the device and play the remote video flow as if it was a local WebCam. This setup worked well and the video was correctly visualized.

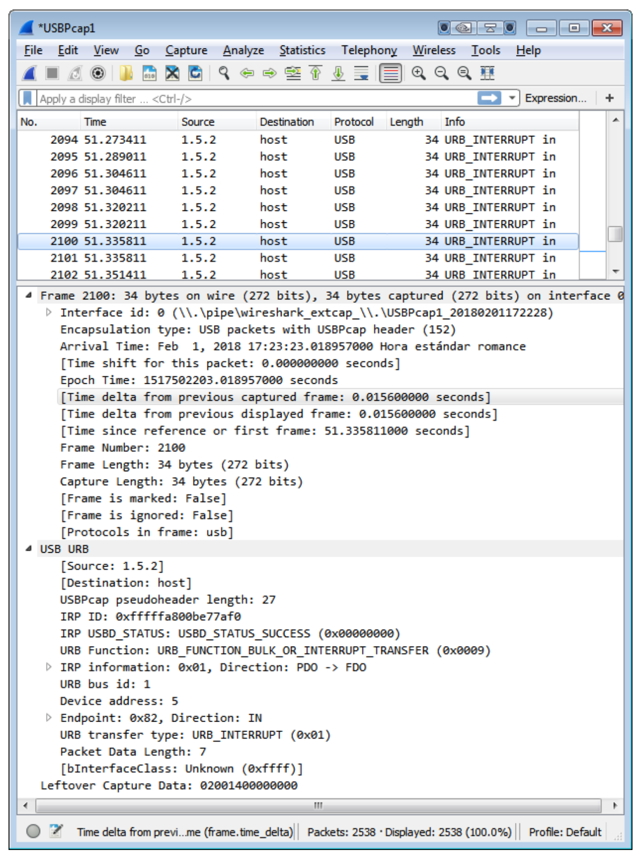


Figure 5. Local USB WebCam data sniffing with Wireshark.

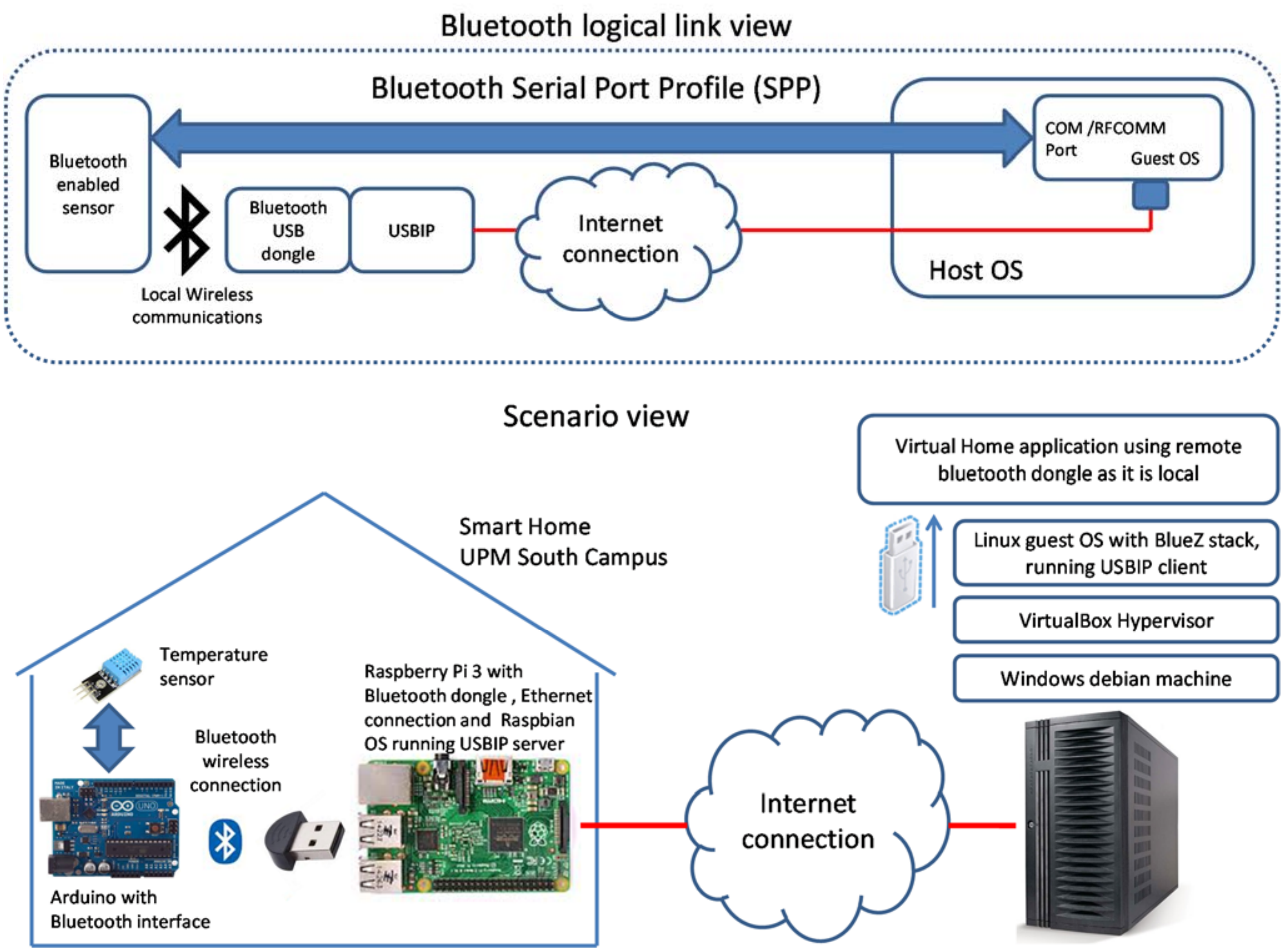


Figure 6. Bluetooth redirection tests setup.

B. *Local Bluetooth Redirection Tests*

The tests described above only show that it is feasible to use remote USB devices as if they were locally connected. We went further with our next experiments, to prove that it is possible to use a Bluetooth USB dongle and have the wireless interface remotely accessible.

To do this we have deployed the setup shown in Fig. 6. The ISP side is emulated by means of a Windows PC equipped with a VirtualBox hypervisor. On top of this virtualization infrastructure, a guest Linux OS runs a USB/IP client interface and the BlueZ Bluetooth stack. All the communications between the sensors and the cloud virtual home control application are Bluetooth.

Any BT (Bluetooth) device must be compatible with at least a subset of the profiles defined by the Bluetooth specification. For example, hands-free systems located in cars have to comply with the Cordless Telephony Profile (CTP) in order to communicate with mobile phones. In our case, the Serial Port Profile (SPP) is used. It emulates a serial cable to provide a simple substitute for existing old RS-232 serial lines. SPP defines how to set up virtual serial ports and connect two Bluetooth enabled devices. From the guest OS point of view, the endpoints of the SPP profile are seen as COM/RFCOMM ports.

Similarly to the USB redirection tests described before, the first local wireless redirection test we have carried out is purely functional. After the pairing phase, the remote Bluetooth USB dongle is perceived as local at the ISP side and seen as a RFCOMM port by the applications. The BT dongle inside the Smart Home receives periodical temperature measurements taken by a sensor that is connected to an Arduino board. These measurements are available at the ISP side thanks to the USB tunneling mechanism. In fact, in order to get the readings it is enough to display the arriving data using a common “cat” command as shown in Fig. 7.

In order to get a better understanding of what the user perception would be in a smart home scenario, we have also carried out a set of round trip delay measurements. Again, an Arduino board with a Bluetooth interface communicates with a control application through a USB dongle. We have tested three networking scenarios: local, same network and Madrid-Alcalá (see IV.A). In the first one, both the USB dongle and the application are located in the same machine. In the other two cases, the traffic of the USB dongle is virtualized and transported to the cloud by means of USB/IP tunnels.

Since what we want to assess are the effects of virtualizing the wireless home interface, the control application just responds back in a ping/pong communication schema. More precisely the procedure is as follows:

```
Round Trip Delay Measurement procedure

While( 1 hour )
        Arduino: set a digital pin up
        Send a ping signal through BT interface
        Wait for response pong
                Arduino: set the digital pin down
        End Wait
        Wait 1 second
End While
```

This way, we obtain a Pulse Width Modulation (PWM) logical signal on the Arduino digital pin that is set up and down by the algorithm above, where each high time corresponds to the round trip time (RTT) of one inquiry. We have arranged for this PWM signal to be read by an oscilloscope and the relevant data stored for further analysis.

Taking one measurement per second for an hour gives us 3,600 round trip times, from which approximate statistical behavior can be estimated. The results are depicted in Figure 8. Fig. 8 shows the raw results of the measurements, whereas Fig. 9 represents the estimated probability density function (PDF) of the round trip time observed in each networking scenario. In addition, Table II below summarizes the main statistical indicators of each RTT measurements series.

By comparing the RTT measurements of the "local" and "same network" scenarios, it is clear that there is no significant difference in their mean values and dispersion. The conclusion is that the delay overhead introduced by the USB/IP tunneling itself is negligible. The vast majority of these RTTs are between 15 and 50 ms.

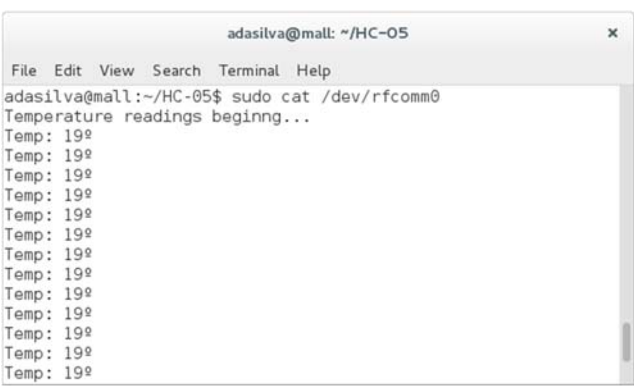


Figure 7. Common serial terminal running in Linux guest OS showing temperature readings.

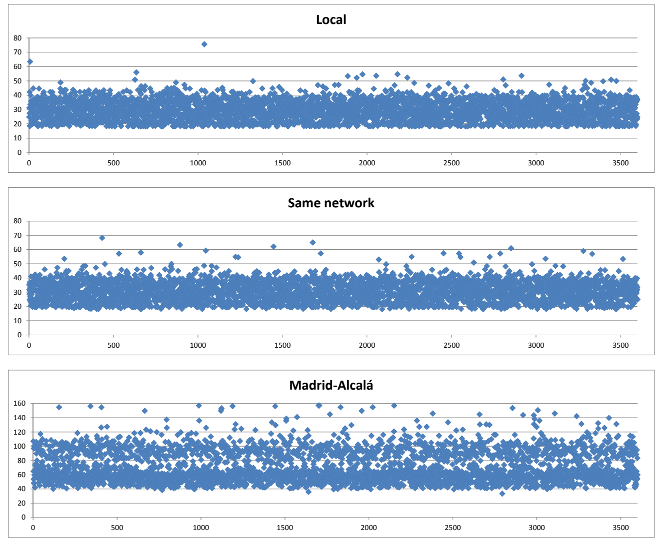


Figure 8. Round trip time measurements in ms.

In the Madrid-Alcalá scenario, the RTTs are considerably higher, ranging in their majority between 40 and 120 ms. Dispersion of these values is also bigger than in the other two cases.

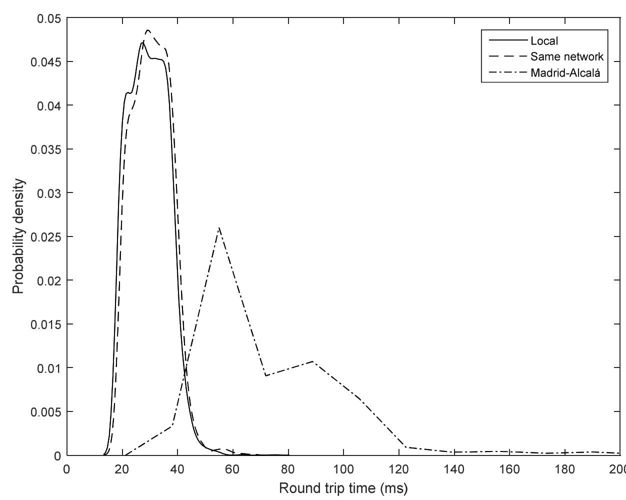


Figure 9. Estimation of the round trip time PDF of each scenario.

TABLE II. STATISTICAL INDICATORS OF RTT MEASUREMENTS (IN MILLISECONDS) FOR EACH LOCAL-REMOTE CONFIGURATION

	Local	**Same network**	**Madrid-Alcalá**
Max	75.6	68.2	1682.3
Min	18.1	18.1	33.5
Average	29.5	30.8	74.1
Std. deviation	6.9	6.9	43.5

Whether these RTT values are acceptable or not depends on the specific IoT application. For applications such as the usual non-critical in-home automation and control, these round trip delays are perfectly acceptable. Indeed, in [5] several applications of the "smart buildings and living" category are considered (among them, building condition monitoring, anomaly detection, lighting control, energy & water use and indoor climate control), and their tolerance to delays range from a few seconds up to a minute. In fact, each of the RTT tests we have performed simulates a common situation by which an event raised in the smart home environment (e.g., a temperature reading) is sent to the cloud, where it is processed and, as a result, an actuation order is generated in response and sent back home (e.g., an order to modify the settings of the HVAC system). This constitutes an example of what can be considered an indoor climate control service.

For more time-sensitive applications (such as alarms that have to be raised very shortly after the triggering event has occurred), it might be necessary to run part of the functionality inside the HRH as a lightweight container.

V. Conclusion and Future Work

This paper describes a basic implementation that shows that the HRH strategy is feasible through the tunneling of USB blocks. The proposed approach, based on common hardware and open software, has several benefits such as its low-cost, low-complexity, easy programmability and alignment with some of the current networking virtualization trends.

We have obtained both functional and non-functional indicators to assess the feasibility of our proposal. It is possible to virtualize and execute remotely basic applications that work as if the USB sources of information were connected locally to the same device. The tests we have made with a USB pen-drive, a conventional WebCam and a Bluetooth USB dongle prove this.

Moreover, USB/IP tunneling, even when combined with a local-remote scenario in which an actual residential network is involved, does not introduce a round trip delay that would impede in any way the deployment of conventional (non-critical) residential control and automation applications.

However, several important issues need further research in order to improve some aspects, for instance the management of uplink communications bandwidth between the HRH and the remote virtualization infrastructure, and the security considerations of a wide scale deployment.

As future work we aim to implement a complete prototype of the proposed architecture and its seamless integration in an existing SDN/NFV infrastructure including its standardized control plane protocols. This will help to assess, among other things, how the control plane affects the initial and subsequent delays observed by the applications, and whether the establishment of the USB/IP tunnels is feasible with the current SDN state of the art protocols. As regards the distribution of the virtualized functions, and given the specific QoS requirements and traffic characteristics of video traffic, we consider video preprocessing an ideal candidate to be runtime deployed in the HRH Docker container as a VNF. This way, if the overall composed video processing service can be divided into modules with clear interfaces, it could be easily deployed across the NFVi infrastructure, including the HRH. In our opinion, the demonstration of this scenario would constitute an interesting proof of concept mixing the service chaining capabilities often mentioned for composite virtualized services and the seamless distribution of components, that is one of the basis of fog computing.

Internet of Things has important security considerations, especially regarding data privacy. This is undoubtedly very important in the case of residential IoT, where concrete measurements are taken inside home, and it becomes even more critical if those data are sent to an external entity (the ISP), as happens in our architecture. Thus, confidentiality and authentication mechanisms have to be in place in any real deployment, and both vendors and NFV providers have to collaborate to create a secure and protected environment with privacy guarantees for users. This trustworthy ecosystem, necessary in our proposal, is not very different from what is expected of existing cloud-based IoT solutions, although it adds an additional stakeholder to the user and the IoT vendor: the VNF provider (or ISP).

On the other hand, the introduction of VNF and SDN brings both security challenges and enhancements [23]. On the positive side, time to upgrade the software with security patches, service availability, incident response time and real-time scaling are all factors that can be enhanced by executing the functionality at the ISP side, since the economies of scale allow for faster reaction to possible attacks and for more efficient and sophisticated preventive actions. In fact, the security functions themselves can be virtualized and executed in a manner that is independent of the specific hardware that enforces them. On the negative side, new threats linked to the virtualization architecture appear, such as hypervisor vulnerabilities (that could allow an attacker to modify some of the VNFs) and specific attacks to SDN controllers.

The performance of the proposed solution will also be subject to more study. Several use cases that include diverse types of services with different traffic requirements will be implemented and tested. Some parameters for the analysis will be: network latency, bandwidth and computing requirements.

Acknowledgment

This work has been supported by Spanish Ministerio de Economía y Competitividad under the grant TEC2015-67834-R.

References

[1] A. da Silva Fariña, A.B. García Hernando, and M.L. Mouronte López, "Residential Wireless Interfaces Virtualization: a Feasibility Study", The Thirteenth International Conference on Wireless and Mobile Communications (ICWMC 2017), Nice, France, July 23 - 27, 2017, pp. 67-72, ISBN: 978-1-61208-572-2.

[2] ETSI GS NFV 002 V1.2.1, "Network Functions Virtualisation (NFV); Architectural Framework", Dec. 2014.

[3] ETSI GS NFV-SWA 001 V1.1.1, "Network Functions Virtualisation (NFV); Virtual Network Functions Architecture", Dec. 2014.

[4] Open Networking Foundation, "Software-Defined Networking (SDN) Definition", 2018. [Online]. Available from: https://www.opennetworking.org/sdn-definition/ 2018.05.22.

[5] J. Mocnej, A. Pekar, W. Seah, and I. Zolotova, "Network Traffic Characteristics of the IoT Application Use Cases", Technical Report ECSTR 18-01, Technical Report Series (ISSN 1179-4259), School of Engineering and Computer Science, Victoria University of Wellington, 6 January 2018. [Online]. Available: https://ecs.victoria.ac.nz/foswiki/pub/Main/TechnicalReportSeries/IoT_network_technologies_embfonts.pdf 2018.05.22.

[6] T. Zachariah et al., "The Internet of Things Has a Gateway Problem", In Proceedings of the 16th International Workshop on Mobile Computing Systems and Applications (HotMobile '15). ACM, New York, NY, USA, 2015, pp. 27-32.

[7] Etisalat, "Evolving the service provider architecture to unleash the potential of IoT". [Online]. Available from: http://www.etisalat.com/en/system/docs/whitepapers/Etisalat-IoT.pdf

[8] Open Networking Foundation, "Specifications". [Online]. Available from: https://www.opennetworking.org/software-defined-standards/specifications/ 2018.05.22.

[9] R. Morabito, "Virtualization on Internet of Things Edge Devices With Container Technologies: A Performance Evaluation", IEEE Access, 5, 2017, pp. 8835 - 8850.

[10] T. Hirofuchi, E. Kawai, K. Fujikawa, and H. Sunahara, "USB/IP - a Peripheral Bus Extension for Device Sharing over IP Network", 2005 USENIX Annual Technical Conference, pp. 47-60.

[11] ETSI GR NFV 001 V1.2.1, "Network Functions Virtualisation (NFV); Use Cases", May 2017.

[12] N. Omnes, M. Bouillon, G. Fromentoux, and O. L. Grand, "A programmable and virtualized network & IT infrastructure for the internet of things: How can NFV & SDN help for facing the upcoming challenges", Intelligence in Next Generation Networks (ICIN), 18th International Conference on, Paris, 2015, pp. 64-69.

[13] M. Niedermeier and H. De Meer, "Constructing Dependable Smart Grid Networks using Network Functions Virtualization", Journal of Network and Systems Management 24, 2016, pp. 449-469.

[14] R. Yu, G. Xue, V.T. Kilari, V.T., and X. Zhang, "Network function virtualization in the multi-tenant cloud", IEEE Netw., vol.29, no. 3, pp. 42-47, 2015.

[15] A. Berl, H. De Meer, H. Hlavacs, and T. Treutner, "Virtualization in energy-efficient future home environments", IEEE Communications Magazine, 47, 12, 2009.

[16] A. Manzalini, R. Minerva, F. Callegati, W. Cerroni, and A. Campi, "Clouds of virtual machines in edge networks", IEEE Communications Magazine, vol. 51, no. 7, pp. 63-70, 2013.

[17] R. Flores, D. Fernández, L. Bellido, N. Merayo, J. C. Aguado, and I. de Miguel, "NFV-based QoS provision for Software Defined Optical Access and residential networks", IEEE/ACM 25th International Symposium on Quality of Service (IWQoS), 2017.

[18] H. Moens and F. De Turck, "VNF-P: A model for efficient placementof virtualized network functions", 10th International Conference on Network and Service Management (CNSM), 2014, pp. 418-423.

[19] S. Clayman, E. Maini, A. Galis, A. Manzalini, and N. Mazzocca, "The dynamic placement of virtual network functions", NetworkOperations and Management Symposium (NOMS), 2014, pp 1-9.

[20] R. Mortier et al., "Control and understanding: Owning your home network", Proc. IEEE 4th Int. Conf. Commun. Syst. Netw. (COMSNETS), 2012, pp. 1-10.

[21] M. Dillon and T. Winters, "Virtualization of home network gateways", Computer, vol. 47, no. 11, pp. 62-65, 2014.

[22] J. Jo, S. Lee, and J. W. Kim, "Software-defined home networking devices for multi-home visual sharing", IEEE Transactions on Consumer Electronics, vol. 60, no. 3, pp. 534-539, 2014.

[23] A. Dutta, "Security Challenges and Opportunities in SDN/NFV Networks", 2016. [Online]. Available from: http://docplayer.net/65655705-Security-challenges-and-opportunities-in-sdn-nfv-networks.html 2018.05.22.

[24] M. Pattaranantakul, Y. Tseng, R. He, and A. Meddahi, "A First Step Towards Security Extension for NFV Orchestrator", SDN-NFVSec '17 Proceedings of the ACM International Workshop on Security in Software Defined Networks & Network Function Virtualization, 2017, pp. 25-30.

[25] IGR, "Moving to Cloud RAN and the Potential of Edge Data Centers", 2016. [Online]. Available: www.commscope.com/Docs/iGR_cRAN_White_Paper.pdf 2018.05.22.

[26] O. Simeone, A. Maeder, M. Peng, O. Sahin, and W. Yu, "Cloud Radio Access Network: Virtualizing Wireless Access for Dense Heterogeneous Systems", Journal of Communications and Networks, vol. 18, no. 2, pp. 135-149, 2016.

[27] F. Ramalho and A. Neto, "Virtualization at the network edge: A performance comparison", IEEE 17th International Symposium on A World of Wireless, Mobile and Multimedia Networks (WoWMoM), Coimbra, 2016, pp. 1-6.

[28] A. Madhavapeddy and D. J. Scott, "Unikernels: Rise of the virtual library operating system", Queue 11 (December 2013), no. 11, 30:30-30:44.

[29] B. B. Rad, H. J. Bhatti, and M. Ahmadi, "An Introduction to Docker and Analysis of its Performance", International Journal of Computer Science and Network Security, vol. 17, no. 3, pp. 228-235, 2017.

[30] M. Chung, N. Quang-Hung, M. T. Nguyen, and N. Thoai, "Using Docker in High Performance Computing Applications", in 2016 IEEE Sixth International Conference on Communications and Electronics, pp. 52-57, 2016.

[31] Y. Huang, A. Stavrou, A. K. Ghosh, and S. Jajodia, "Efficiently Tracking Application Interactions using Lightweight Virtualization", VMSec '08 Proceedings of the 1st ACM workshop on Virtual machine security, 2008, pp. 19-28.

[32] L. Zeltser, "Security Risks and Benefits of Docker Applications Containers", 2018. [Online]. Available: https://zeltser.com/security-risks-and-benefits-of-docker-application/ 2018.05.22.

[33] Docker Inc., "Docker Security", 2018. [Online]. Available: https://docs.docker.com/engine/security/security/ 2018.05.22.

[34] CIS Docker 1.12.0 Benchmark, 2017. [Online] Available: https://www.cisecurity.org/cis-benchmarks/ 2018.05.22.

[35] The European Telecommunications Standards Institute, "Network Functions Virtualisation (NFV). Network Operator Perspectives on NFV priorities for 5G", 2017. [Online]. Available: https://portal.etsi.org/nfv/nfv_white_paper_5g.pdf 2018.05.22

[36] Y. Gittik "Distributed Network Functions Virtualization (White paper)", March 2014.

[37] A. B. García, A. Da Silva, L. Bellido, F .J Ruiz, and D. Fernández, "Virtualization of Residential IoT Functionality

by Using NFV and SDN", In 2017 International Conference on Consumer Electronics (ICCE), Las Vegas, USA, January, pp. 8-10, 2017.

[38] S. Tomovic, K. Yoshigoe, I. Maljevic, and I. Radusinovic, "Software-Defined Fog Network Architecture for IoT", Journal Wireless Personal Communications: An International Journal archive, vol. 92, no. 1, 2017, pp. 181-196.

[39] N. Bizanis and F. A. Kuipers, "SDN and Virtualization Solutions for the Internet of Things: A Survey", 2016, 4, IEEE Access, pp.5591- 5606.

[40] S. M. M. Gilani, T. Hong, W. Jin, G. Zhao, H. M. Heang, and C. Chuan, "Mobility management in IEEE 802.11 WLAN using SDN/NFV technologies", EURASIP Journal on Wireless Communications and Networking, 2017.

[41] S. Abdelwahab, B. Hamdaoui, M. Guizani, and T. Znati, "Cloud of Things for Sensing as a Service: Sensing Resource Discovery and Virtualization", In 2015 IEEE Global Communications Conference (GLOBECOM). IEEE, pp. 1–7.

[42] R. Flores, D. Fernández, and L. Bellido, "A user-centric SDN management architecture for NFV-based residential networks", Computer Standards & Interfaces, Vol. 54, Part 4, November 2017, pp. 279-292.

QoS Assessment and Modelling of Connected Vehicle Network within Internet of Vehicles

Asiya Khan and Sanjay Sharma
School of Engineering,
University of Plymouth, Plymouth, PL4 8AA UK
Email: asiya.khan;sanjay.sharma@plymouth.ac.uk

Cindy Goh[1] and Cheng Leong Lim[2]
[1]School of Engineering, Glasgow University,
Glasgow G12 8LT UK
[2]NXP Semiconductors, Singapore 138628
Email:cindy.goh@glasgow.ac.uk;
chengleong.lim@nxp.com

***Abstract*—Connected vehicles have huge potential in improving road safety and traffic congestion. The primary aim of this paper is threefold: firstly to present an overview of network models in connected vehicles; secondly to analyze the factors that impact the Quality of Service (QoS) of connected vehicles and thirdly to present initial modelling results on Link QoS. We use the open access Geometry-based Efficient Propagation Model (GEMV^2) data to carry out Analysis of Variance, Principal Component Analysis and Classical Multi-Dimensional scaling on the link quality for vehicle-2-vehicle (V2V) and vehicle-2-infrastucture (V2i) data and found that both line of sight and non-line of sight has a significant impact on the link quality. We further carried out modelling using system identification method of the connected vehicle network (CVN) in terms of Link QoS based on the parameters identified by the QoS assessment. We evaluated the CVN in terms of a step response achieving steady-state within 80 seconds for V2V data and 500 seconds for V2i data. The work presented here will further help in the development of CVN prediction model and control for V2V and vehicle-2-anything connectivity.**

Keywords-QoS; IoV; ANOVA; PCA; CMD; CVN; V2V; V2x;

I. INTRODUCTION

The autonomous car phenomenon is underway in most developed economies. While we are many years away from full autonomy of vehicles, partial autonomy is becoming a reality. Connected vehicles offer huge potential in improving road safety and congestion. Authors in [1] have conducted quality of service (QoS) analysis on connected vehicle network which has been extended in this paper. According to a recent report from the Department of Transport, from October 2015 to September 2016, there was around 183,000 casualties resulting from traffic accidents of which 1,800 were fatal and over 25,000 were life changing [2]. Therefore, the vision is that vehicle-2-anything (V2x) connectivity will reduce this figure by at least 76%. A number of developed countries are trialling fully and semi-autonomous cars on the road. Google's cars have driven 1.2 million miles in USA, with Germany, China and the UK, also looking to open trials. Connected vehicles will play a key part in traffic management of autonomous cars. Within the next five years there will be some form of autonomous driving on the roads of UK.

Connected vehicles are defined as a set of moving networked computer systems with dozens of electronic control units (ECUs), hundreds of sensors and million lines of code [3]. Research investigating the suitability of wireless channels is a significant starting point to them becoming a reality in the near future [4][5]. This will also help towards the modelling of wireless channels for connected vehicles. The benefits of vehicle-2-vehicle (V2V) connectivity especially in areas of collision avoidance and congestion management are huge, V2V is becoming a reality and automobile industry is currently working towards standardization.

The emerging Internet of Vehicles (IoV) is offering the platform to provide real time exchange of information to realize the opportunity of improving road safety and congestion. It has huge applications in autonomous car revolution, intelligent transportation system and smart city. IoV integrates three networks – an inter-vehicle network, an intra-vehicle network and vehicular mobile Internet. Therefore, IoV integrates these three networks and is defined as "a large-scale distributed system for wireless communication and information exchange between V2x according to agreed communication protocols and data standards" [6].

There are a number of challenges within the IoV network based on the priority of data exchange messages. For example, priority has to be given to safety critical messages, whereas on-board messages related to infotainment will be lower on that scale. The work presented in [7] proposes an abstract network model for IoV based on individual and swarm activities. Petri-nets have been used recently in vehicular authentication [8], modelling and control of vehicular networks [9] and traffic signal analysis in [10]. The work presented in [11] models vehicular networks using spatio-temporal locality and information-centric networks (ICN) are presented in [12] to model the connected networks. Recently, the concept of Network of Things (NoT) with Internet of Things (IoT) has been presented in [13].

A number of researchers have presented findings both on technique [9] and a network model [14]. Petri nets are proposed in [15] for such time critical distributed communication and control systems. GEMV^2, a geometry-based V2V channel model has been presented in [14], which measures link quality by factoring outlines of vehicles, buildings, and foliage to distinguish between the three types of links; the links are Line of Sight (LOS), Non-LOS due to vehicles and Non-LOS (NLOS) due to static objects. In addition, the link quality is calculated with the large-scale signal variation deterministically and the small scale-signal variation stochastically based on the number and size of surrounding objects. $GEMV^2$ is freely available to be used by researchers.

The aim of this paper is to identify and present those challenges and opportunities associated with Quality of Service (QoS) in connected vehicles and to identify the modelling direction for Connected Vehicle Network (CVN) by conducting Analysis of Variance (ANOVA), Principal Component Analysis (PCA) and Classical Multi-Dimensional scaling (CMD) on the factors that impact link QoS. We further apply the concept of NoT to the emerging IoV as presented by NIST [13] and review the connected network models presented in literature identifying the challenges and solutions. Here, we define CVN as the network between V2V and V2x and where the position/velocity of the vehicle is predicted from the previous vehicle/x. The 'x' in V2x represents vehicle/infrastructure/roadside sensors/anything else deemed suitable. The vision for CVN is that each vehicle on the road will be able to communicate with other vehicles and this set of data and communication will support a new generation of active safety applications and systems [15]. Wireless technologies and their potential challenges in providing vehicle-to-x connectivity are presented in [5]. An overview of applications and associated requirements of vehicular networks are presented in [16]. Internet mobility in vehicular scenarios along with their challenges is presented in [17]. With ever increasing connectivity and a vision that migrates towards smart cities, security issues and the challenges such as propriety networks, inter-operability between networks, etc. therein are immense. The work in [18] presents some of the security challenges in vehicular ad hoc networks (VANET), whereas [19] focuses on the four working groups on scientific foundations of vehicular networking and presents their findings. Connected Vehicle Network is modelled using a black-box approach that comprises of vehicles with wireless V2V communication using link length estimator to identify the number of vehicles in the network [20], whereas [21] presented modelling of future state of a vehicle in a platoon based on preceding vehicle position and velocity.

In this paper, we use the data from $GEMV^2$ to carry out ANOVA, PCA and CMD. Doing so, helps us to better understand the QoS relationship between the link QoS and the factors that impact it. We chose four factors that impact link quality as Line of Sight (LOS), Non-Line of Sight (NLOS), number of neighbours per vehicle (neighbours) and the number of neighbouring vehicles whose received power was above the threshold (neigh-thresh). Based on the QoS assessment [1], we model the parameters to predict the Link QoS using System Identification method [22]. The parameters are described in Section III.

The work presented in this paper differs from the ones listed above since it provides an in depth analysis on the various wireless channels available for connected vehicles based on our QoS assessment of the $GEMV^2$ data.

The rest of this paper is organized as follows. Section II formulates the problem statement, whereas, Section III gives an overview of the network models for connected vehicles. In Section IV the channel requirements for CVN is presented. Section V describes the QoS assessment on $GEMV^2$ data, whereas, Section VI presents the modelling of Link QoS. Section VII discusses the research challenges in CVN modelling. Conclusions and future work are presented in Section VIII.

II. Problem Statement

The sheer volume of traffic leads to congestion during (increasing long) peak periods, and high traffic density increases the probability of collision. If each vehicle in the system is a node in a communication system, then drivers can be provided with easier warning of impending issues. This IoV would enable dynamic planning in the event of local constrictions in traffic flows. These systems are equally applicable to drivers and to autonomous vehicles. Successful implementation of such systems should lead to shorter journey times, more efficient use of resources (minimized travel time and fuel use), and avoidance of accidental damage and consequent financial loss and human injury/death. These systems need to be resilient and while communication distances are short in heavy traffic the system should be capable of working to the same QoS in the early hours when traffic is sparse. QoS in this context is the minimum acceptable quality of the connected vehicle network to enable V2V or V2x communication type.

The intelligent transport system (ITS) reference architecture from [23] has been adapted and is presented in Figure 1. It is a protocol stack inspired from the Open Systems Interconnection (OSI) model and defines three layers as 'access', which will support the wireless access networks/wireless channels, a network & transport layer which supports the routing protocols, data transfer, etc. Above it sits the facilities layer, which will support the application/information. Here, we define the position/velocity of the vehicle in this layer. The application layer supports vehicle operations based on parameters of reliability, security, latency, etc. measured in terms of LOS, NLOS, etc. The layers of application, management and security run across both horizontally and vertically and provides cross layer commands and information.

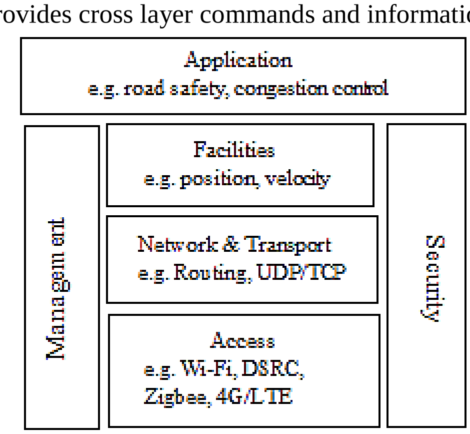


Figure 1. ITS reference traffic structure (adapted from [23])

QoS of this IoV is affected by a number of parameters. These parameters can be divided in access network, facilities and application levels. In the access network layer QoS can be characterized as:

$$QoS = f(PER, delay, jitter, latency, \ldots\ldots)$$

In the facilities level, QoS is given as:

$$QoS = f(Vehicle\ speed, Vehicle\ location)$$

In the application level, QoS is given as:

$$QoS = f(Vehicle\ length, LoS, NLoS, \ldots\ldots)$$

Figure 2 shows the overall modelling direction combining parameters from all three layers.

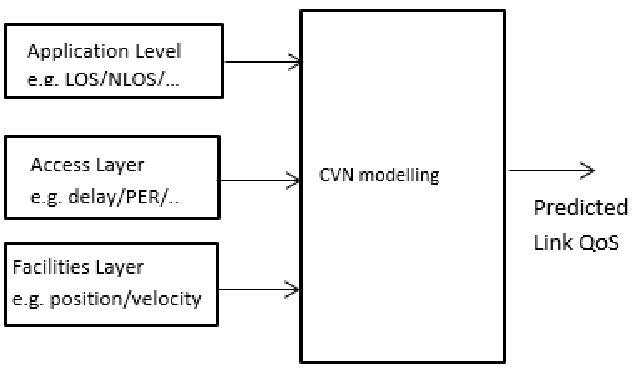


Figure 2. Prediction of Link QoS

Therefore, the contributions of the paper are three-fold:

- to present an overview of network models and wireless channel requirements in connected vehicles.
- to present ANOVA, PCA and CMD on GEMV^2 data to understand the impact of line of sight, non-line of sight, neighbours and neigh-thresh per vehicle on link quality.
- to present initial modelling results on Link QoS.

III. AN OVERVIEW OF NETWORK MODELS IN CVN

This section presents an application of NoT to IoV and an overview of the two network models [24] presented in literature.

A. *NoT applied to IoV*

The concept diagram of connected vehicles is presented in Figure 3, which illustrates V2V and V2x connectivity using various access networks which is in turn connected to the core network. The concept behind Figure 3 is that connected vehicles will be able to communicate with each other and with an intelligent transport system (ITS) using different wireless channels such as Wi-Fi, 4G/LTE, etc. QoS in such application will be critical as vehicles come out of one network into the other especially at handover points. Connected vehicles are the building blocks of emerging Internet of Vehicles (IoV) and Network of Things (NoT) [13], which is defined on five primitives as sensors, aggregator, communication channel, external utility and decision trigger. All vehicles or 'x' will have sensors connected that will be able to transmit/receive 'useful' information. This information is converted by an aggregator, defined as a mathematical function implemented in software that transforms raw data into some 'useful' meaning. This is underpinned by the communication channel, e.g., WiFi, 4G, etc. The external utility can be a software/hardware and will execute processes into the overall workflow of NoT. Finally, the decision trigger creates the final result needed to satisfy the requirements of NoT.

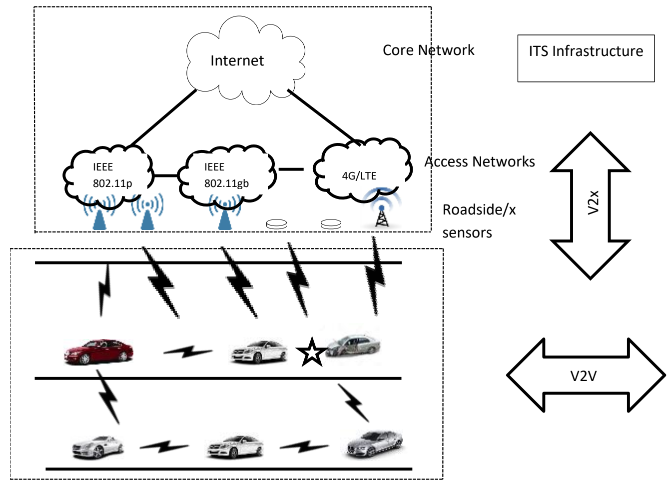


Figure 3. V2V and V2x concept diagram

Within the NoT [13] all vehicles will have sensors connected that will be able to transmit/receive 'useful' information. This information is converted by an Aggregator, defined as a mathematical function implemented in software that transforms raw data into some 'useful' meaning. Both Sensor and Aggregator are shown as Roadside sensors in Figure 3. This is underpinned by the communication channel, e.g. WiFi, 4G, etc. Again, Figure 3 shows the wireless channels such as Wi-Fi/4G etc. between V2V and V2x. The External Utility can be a software/hardware and will execute processes into the overall workflow of NoT. Finally, the Decision Trigger creates the final result needed to satisfy the requirements of NoT. The External Utility and Decision Trigger is combined together and presented within ITS in Figure 3.

TABLE I. IOV PRIMITIVES

NIST Primitives	Proposed Primitives	Feature
Sensor	Sensing Technologies	Wireless and wired, sensors, RFID,
Aggregate		
Communication Channel	Communication Channel	DSRC/Wave, Zigbee, Bluetooth, Wi-Fi, 4G/LTE
External utilities	Data Processing	Data created by connected vehicles, and how it is processed
Decision Trigger		

Based on these NoT primitives [13], we present three primitives. We combine the primitives of Sensor and Aggregator as just Sensing Technologies, Communication Channel and again combine External Utility (eUtility) and Decision Trigger as one and call it Data Processing as shown in Table I. In Table I, feature describes the potential features for each primitive.

B. Swarm and Individual Network Model

The model presented in [7] integrates human, vehicle, thing and environment. The individual model focuses on one vehicle and the swarm model focuses on multi-user, multi-vehicle, multi-thing and multi-network scenarios. Through swarm intelligence, crowd sensing and sourcing and social computing, IoV can provide services/applications. Factors such as network partitions, route failures, change in channel quality and data rate and network load are addressed using swarm intelligence computing at the service providing stage. This is shown in Figure 4.

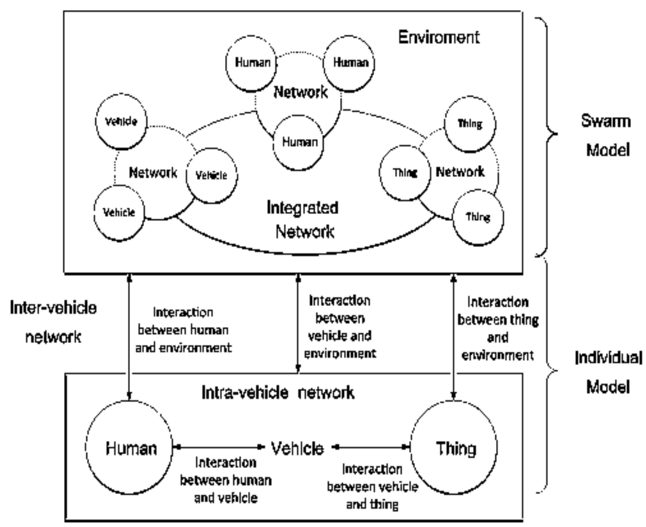


Figure 4. Swarm and Individual Network Model of IoV [7]

Authors in [7] also highlight that understanding the service limits is critical for sustainability, i.e. network resources under diverse high-dimensional data and limited bandwidth of the wireless network.

C. Cloud, Connection and Clients

Three major network elements of IoV are identified in [25] as cloud, connection and client as shown in Figure 5. The 'cloud' infrastructure provides a platform for a range of wireless access technologies. With the magnitude of traffic related information likely to drastically increase, it is ideal to handle the information using cloud computing framework. 'Connection', on the other hand, utilises Third Party Network Inter Operator (TPNIO) to reduce direct Service Level Agreement (SLA) between the operators of the networks, enabling seamless roaming without compromising the quality and security of network operators. The 'client' element with the help of Wireless Access Technology (WAT) are broadly prioritized and split applications into safety and management oriented and business oriented.

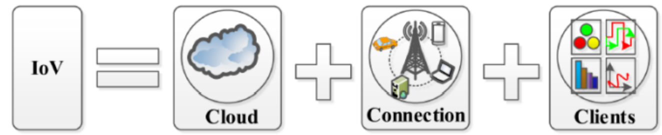


Figure 5. The three network elements of IoV [25]

D. Summary of Network Models for CVN

The challenge for any network model in IoV is to be able to exchange information from V2V and V2x, where x can be a roadside sensor, another device or a person. In addition, there may be incompatibility among devices, different qualities and response time for Internet connections and limited access to data processing and storage. There will be additional complexity where some vehicles will be connected while others not.

Future and emerging vehicle applications will consume a huge amount of sensor data in a collaborative manner. Content centric [26] and information-centric networks [12] will play a key role. Vehicles move fast, therefore, in a content-centric networking style, vehicle position, speed and direction from the rest of the vehicles are continuously sent. Whereas, ICN focusses on what instead of where to fulfil primary demands from both content publishers and consumers. Vehicular-cloud and ICN will contribute to the 'cloud' to produce advance vehicular services, resource sharing and storing. Four categories of services are provided by cloud computing as - Software as a Service (SaaS), Platform as a Service (PaaS), Network as a Service (NaaS) and Infrastructure as a Service (IaaS). SaaS is mainly application working over the Internet, whereas, PaaS provides a platform to build application and services, virtual network are provided by NaaS to the users and IaaS provides computation and storage services. The proposed architecture for ICN – Named Data Networking (NDN) [27] has been extended to vehicular networks where content is found and not hosts or IP addresses.

The revealing of location information has huge concerns in vehicle privacy. In addition, location verification of neighbouring vehicle is also challenging due to the absence of trusted authority in vehicular communication. To capture vehicles in line of sight and away from sight presents yet another challenge due to the impact of moving and static obstacles in the network model. The integration of automotive and information technology will be promoted as a result of IoV. The biggest challenge in IoV implementation is the lack of coordination and communication. This paper aims to address the QoS issues in communication challenge. Some of the challenges identified are:

- Maintaining an accurate line of sight
- Accounting for vehicles/x that are outside the line of sight

- Position/velocity of the vehicle in order to model the dynamic platoon of vehicles
- Vehicles that are not connected
- Security considerations and protection from theft
- Integration of different wireless protocols e.g. DSRC, IEEE 802.11abgn Wi-Fi, 4G/5G cellular networks, VLC
- Device-to-Device (D2D) communication (defined as direct communication between devices in range proximity without the involvement of a network infrastructure) [28] based on LTE
- Safety vs comfort applications
- Integration with cloud architecture
- Big data analysis in IoV
- QoS guarantee – investigate into software defined networking techniques based on the combined information from multiple sources rather than individual

IV. CVN Channel Requirement

A number of applications ranging from infotainment, for example, media downloading to traffic safety applications, such as driving assistance co-operative awareness impose diverse requirements on supporting vehicular networking technologies. There will be a huge emphasis on inter-networking between the different standards in order to achieve seamless communications. In addition, there are different requirements for inter-vehicle (V2V or V2x) and intra-vehicles networks. Intra-vehicle is defined as all the ECUs within the vehicle communicating to the driver and includes infotainment. Hence, all the wireless channels described in this section may play a role in the connected vehicle application. Therefore, this section provides an overview on the wireless channels available and the connectivity challenges required in a V2V or V2x communication type.

A. DSRC/Wave

Dedicated short-range communications with wireless access in vehicular environments (DSRC/WAVE) as defined by IEEE 802.11p and IEEE1609 (higher layer standard based on IEEE 802.11p) is a key enabling wireless technology for both V2V and V2R communications. DSRC works in 5.9GHz band with a bandwidth of 75MHz in the US and 30MHz in Europe and an approximate range of 1000m. It is designed for both one way and two way communication. DSRC are not compatible in Europe, Japan and US. Currently, DSRC is the default broadcast communication protocol used. Some limitation of DSRC includes its dedicated spectrum in supporting V2V communication type [29] and lack of QoS support. Key application for DSRC is roadside sensors, which transmit information about hazardous conditions, road surface and distance between vehicles and anti-collision information.

B. Zigbee

Zigbee is based on IEEE 802.15.4 specification intended for wireless personal area network applications with low power and cost. Zigbee also has applications in V2R connectivity where the moving vehicle exchanges information with the roadside sensors [30]. The Zigbee enabled roadside sensors then updates traffic status to an intelligent control system seamlessly. It also has application in intra-vehicle networking where a small wireless sensor network is established between the sensors.

C. Visible Light Communication (VLC)

The use of visible light communication (VLC) for V2R communication is proposed in [31]. VLC is defined by IEEE 802.15.7 standard and can support data rate up to 96Mb/s through fast modulation of LED light sources [29]. It is an emerging area of research given the possibility of augmenting existing infrastructure such as traffic lights. However, one key limitation of VLC is any poor weather conditions such as rain and fog could ultimately degrade its communication reliability.

D. Wi-Fi

Wi-Fi standards are based on IEEE 802.11 series, mainly using the 2.4/5GHz band. A number of automobile manufacturers are building new cars with in-built Wi-Fi capability, providing infotainment applications. V2V connectivity could also foster the integration of bicycles and pedestrians into the networks [16] using Wi-Fi. This has a huge potential in improving road safety and reducing the number of accidents as a result of blind spots.

TABLE II. Summary of Wireless Channels for V2V and V2x Communication Types

Wireless Channels	Advantages	Disadvantages
DSRC/WAVE	Default broadcast network currently used	Limited coverage, (~1000m), QoS not supported
Zigbee	Mesh network, scalable, no need for centralized control	Low and limited data rate, not mature security, limited coverage (10-100m)
VLC	Infrastructure already there, 1-2000m range	Early stages/cost of conversion
Wi-Fi	Widely implemented, 35m indoor and 115m outdoor	Interoperability with other protocols
4G/LTE	Existing infrastructure, several Km range	Interoperability with other protocols

E. 4G/LTE

Long-Term Evolution (LTE) is a standard for high speed communications for mobile phones and data terminals. The standard is developed by 3GPP. The key advantage of LTE-connected cars [4] is having cars connecting directly to the Internet through existing 4G-LTE cellular network. Work in [32] presents a hybrid scheme that can achieve seamless IP communication over mobile Internet access.

F. *Summary of CVN Channel Requirements*

Table II summarizes the various wireless channels, their standard requirements and potential advantages and disadvantages for V2V and V2x. The current industry trends are choosing DSRC and 4G/LTE as the best way to offer connectivity between cars. Many critical applications are linked to safety applications, e.g., air bag control, automatic braking, etc. Inter-operability between these networking standards will be an important milestone. The work presented in [33] concludes that DSRC configuration choice has an impact on safety messages successfully transmitted. In addition, as suggested in [34][35], an upper limit on information provided to the vehicle may be necessary to prevent overloading drivers with information. This poses additional requirements and challenges towards the standardization of wireless channels for vehicle communication. Depending on the communication type, e.g., V2V or V2x, all of the wireless channels presented in Table II will be relevant and the CVN modelling has to take that into account.

V. QoS Assessment in CVN

This section presents the QoS assessment using Analysis of Variance (ANOVA), Principal Component Analysis (PCA) and classical multidimensional scaling (CMD) in MATLAB on GEMV^2 data for V2V and V2I. CMD has been introduced as an extension to the analysis conducted in [1] to further understand and confirm the results of the interactions. This will help us in understanding the interaction between the four parameters chosen and their impact on the link quality and lay the foundation in establishing the modelling direction for CVN.

A. *GEMV^2*

GEMV^2 (Geometry-based Efficient Propagation Model for V2V communication) [13] data is freely available and is implemented in MATLAB. GEMV^2 measures large-scale variation calculated deterministically and small-scale signal variation stochastically based on the number and size of the surrounding objects. Both the signal variation is measured in decibels.

We use the GEMV^2 data of large-scale and small-scale signal variation under the influence of four different conditions - they are LOS, NLOS, the number of neighbouring vehicles and the neigh-thresh per vehicle. The data is available for both V2V and V2I. The communication channel is IEEE802.11p.

LOS links have an unobstructed path between communicating vehicles, whereas NLOS is obstructed by vehicles and buildings. Neighbours is defined as the number of transmitting vehicles in the network and neigh-thresh is defined as the number of neighbouring vehicles whose received power was above the threshold.

B. *ANOVA on GEMV^2 Data*

ANOVA was carried out on the GEMV^2 dataset. ANOVA is chosen as it enables us to understand the interaction between the four parameters on link quality. Table III presents the results. ANOVA was carried out on large-scale signal variation only for both V2V and V2I as the interaction between parameters was found to be not as significant for small-scale variation.

Table III. ANOVA Results for Main and Interaction Effects for V2V & V2I Data

Source	Sum of Squares	Degree of freedom	Mean Squares	F-statistics	p-value
V2V					
LOS	23646.7	38	622.283	5.37	0
NLOS	18100	39	464.102	4	0
Neighbours	6377.9	41	155.558	1.34	0.0828
Neigh-thresh	189.3	4	47.321	0.41	0.8028
LOS*NLOS	66.9	1	66.9451	17.73	0.0007
LOS*Neighbours	141	3	47.0094	12.45	0.002
NLOS* Neighbours	24.6	1	24.6413	6.52	0.0212
Neighbours* Neigh-thresh	34.4	1	34.3572	9.1	0.008
V2I					
LOS	340.4	7	48.625	2.5	0.0181
NLOS	12669.6	20	633.479	32.63	0
Neighbours	60.3	7	8.614	0.44	0.8733
Neigh-thresh	1248.2	6	208.027	10.72	0
LOS*NLOS	69.4	2	34.71	0.52	0.6244
LOS*Neighbours	0	2	0.017	0	0.9998
NLOS* Neighbours	33	14	2.357	0.04	1
Neighbours* Neigh-Thresh	0	1	0	0	0.9995

Tables III shows the results of the ANOVA. The p-value is derived from the cumulative distribution function of F [36] and a small p-value indicates that the link quality is significantly influenced by the corresponding parameter. Between V2V communications, both LOS and NLOS have significant impact on the link quality, whereas between V2I communications, NLOS is slightly more significant than LOS and the Neigh-thresh have a higher impact on link quality. However, for V2V, all four parameters have small p-values indicating that they all in varying degree are significant. However, it is interesting to note that, in V2I, the number of neighbours per vehicle is not that significant. For V2V, the combined interaction between LOS and NLOS and NLOS and Neighbours is most significant. Whereas, for V2I, the combined interactions are less significant compared to the individual. To better understand the interactions, PCA investigation is carried out.

C. *PCA on GEMV^2 Data*

PCA was chosen as it reduces the dimensionality of the data while retaining as much information as possible. PCA involves calculating the eigenvalues and their corresponding eigenvectors of the covariance or correlation matrix. The covariance matrix is used where the same data has the same set of variables and correlation matrix is used in the case where data has a different set of variables. In this paper, the covariance matrix was used because of the same dataset.

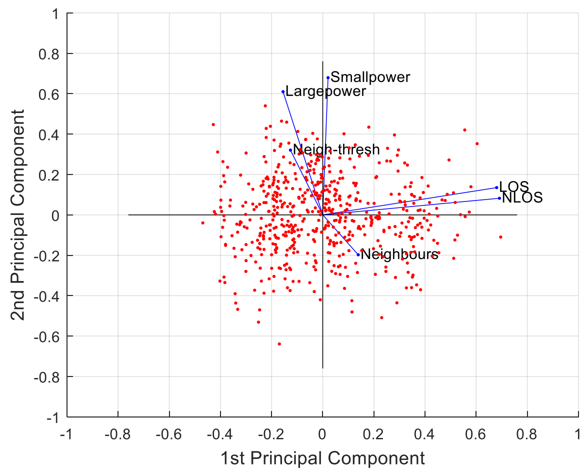


Figure 6a. PCA results for V2V

Figures 4a and 4b show the PCA results for V2V and V2I respectively. In addition to the four factors, both large-scale (Largepower) and small-scale (Smallpower) signal variation is used. The horizontal axis represents the first principal component and the vertical axis the second. Each of the parameters is represented by a vector. There are six components in Figures 6a and the first three components account for more than 90% of the variance. Figure 4a shows the first principal component contributes largely to LOS and NLOS.

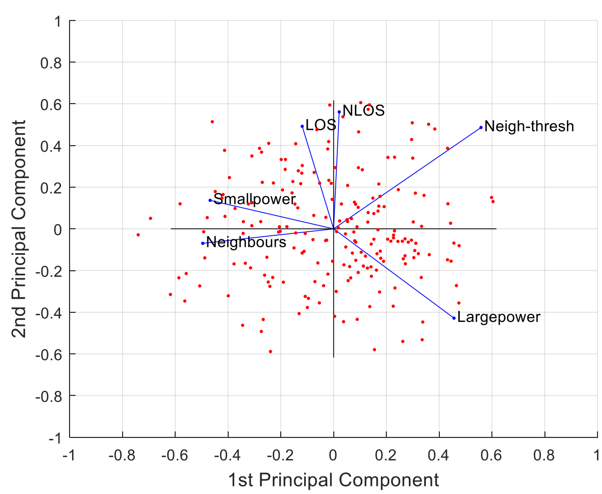


Figure 6b. PCA results for V2I

Figure 6b shows the PCA results for V2I. Similar to Figure 6a, Figure 6b the first three components account for over 80% of the variability. Points on the edge of the plot have the lowest scores for the first principal component.

D. *CMD on GEMV^2 Data*

Classical multidimensional scaling (CMD) was carried out on the GemV^2 data. CMD takes a matrix of interpoint distances and creates a configuration of points. It allows data to be visualized to get a sense of how near or far points are from each other. Therefore, it offers a way of confirming the results obtained from PCA and ANOVA in terms of the interactions of the chosen parameters. A scatter plot of those points provides a visual representation of the original distances and can produce a representation of data in small number of dimensions.

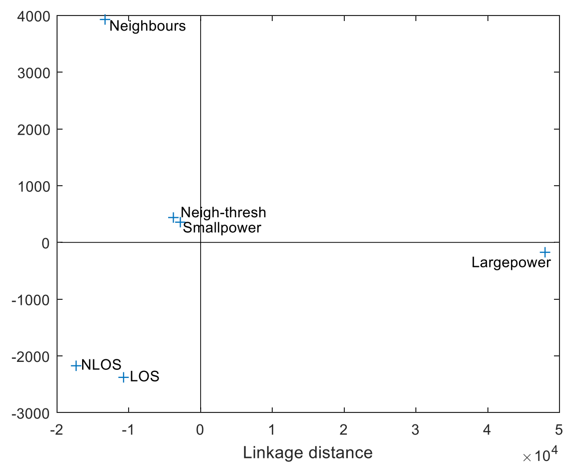


Figure 7a. CMD results for V2V

Figures 7a and 7b show the CMD plot of V2V and V2I data respectively.

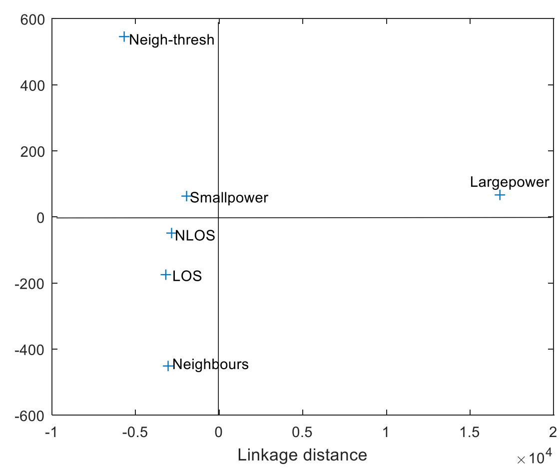


Figure 7b. CMD results for V2I

Comparing Figure Fig 7a to 6a, we get similar results and shows that both LOS and NLOS are closely correlated. Similarly, Figure 7b mirrors Figure 6b and shows close correlation between LOS and NLOS. CMD analysis confirms the results obtained by ANOVA and PCA earlier. The cophenetic coefficient was 98.76% for V2V and 99.6% for V2I.

VI. CVN MODELLING

We modelled the CVN in terms of the Link Quality with the three QoS parameters – LOS, NLOS and neighbours. Figure 8 shows an overview of CVN modelling. Neigh-thresh was not chosen as it was not found to be significant from the QoS assessment.

In the future, we will extend this to include parameters from each layer, e.g., from the application layer, congestion modelled by number of vehicles, LOS, NLOS. Similarly, the access layer is modelled by parameters such as Packet Error Rate (PER) and delay and the facilities layer contributes the vehicles position and velocity.

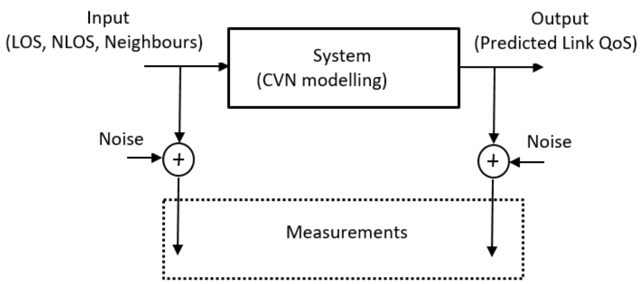


Figure 8. CVN modelling

We used System Identification toolbox in MATLAB on the GEMV[2] data to present the response of V2V and V2i when subjected to a step response. The advantage of this method is that it uses measured data directly to estimate the model. It uses Auto Regressive Exogenous (ARX) models based on the method of least squares to determine the best fit line to the data. The method generalizes to finding the best fit using simple calculus and linear algebra of the form:

$$y = a_1 f_1(x) + \ldots + a_K f_K(x) \tag{1}$$

Where, $f_1,\ldots,f_k$ are given functions to find values of coefficients $a_1,\ldots,a_K$.

"ARX model structure, in discrete time", is a difference equation with the following form:

$$y(t) + a_1 y(t-1) + \cdots + a_{n_a} y(t-n_a) = b_1 u(t-n_k) + \cdots + b_{n_b} u(t-n_k-n_b+1) + e(t) \tag{2}$$

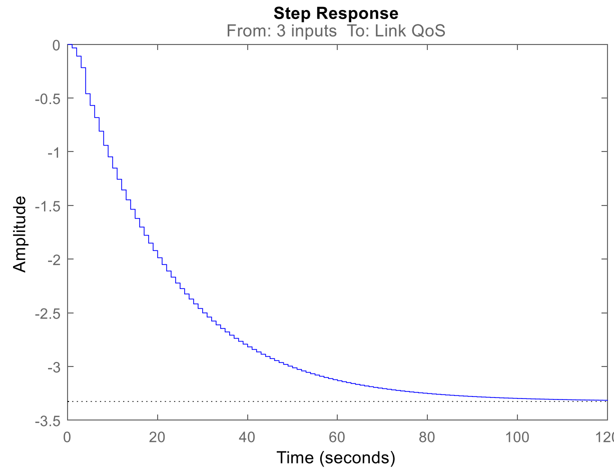


Figure 9a. Step response for V2V data

Where, y(t) it's the output at time t , u(t) is the input at time t, n_a is the number of system poles, n_b is the number of 'b' parameters and 'b' is equal to number of zeros plus 1, n_k is the number of delays in the system. The error function in eq. (2) is given by e(t) and is defined as the white-noise disturbance value and given as noise in Figure 8. The discrete time transfer function can be defined as:

$$H(z) = \frac{Y(z)}{U(z)} = \frac{b_1 z^{-1} + \cdots + b_{n_b} z^{-n_b}}{1 - a_1 z^{-1} - \cdots - a_{n_a} z^{-n_a}} \tag{3}$$

Figures 9a and 9b show the step response of the 3 inputs on the Link QoS for v2v and v2i data.

The step response in Figure 9a shows that the link QoS has a steady state response (~-3.4) with the three inputs chosen for V2V.

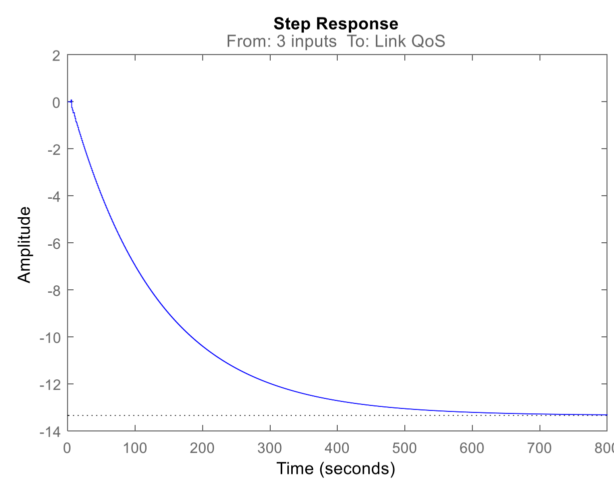


Figure 9b. Step response for V2i data

The step response in Figure 9b shows that the link QoS has a steady state response (~-13.2) with the three inputs chosen for V2i. This enables us to extend our inputs in the future into other layers as shown in Figure 2 to give a better prediction for the CVN model.

VII. Research Challenges in CVN

Our small-scale QoS assessment highlighted some of the research challenges and hence potential opportunities for further work are as follows:

(i) Overcoming QoS issues in connected vehicles is fundamental to the successful deployment of V2x connectivity. The QoS can be affected by networking parameters such as bandwidth, delay and latency. In addition, parameters such as the distance between vehicles, road-side sensors and the speed of the vehicle all play a part towards the QoS of the V2x network thus integrating connected vehicles into IoT ecosystems [37]. QoS will be further divided between V2x service reliability for safety related applications where parameters such as time-sensitivity during message transfer, guarantee of message delivery, etc. are highest priorities. While QoS of on-board applications, e.g., infotainment will be lower in priority.

(ii) We also identified that the needs for trade-off between the amount of intelligence sitting with the vehicle for intra-vehicle connectivity and to that controlled remotely via an intelligent control system. Different wireless channels will be suitable for inter-vehicle vs intra-vehicle connectivity. For example, on-board sensors that can sense a motorbike/bicycle within the blind spot of the driver can greatly improve road safety and reduce accidents.

(iii) Prediction of CVN will be based on information centric network paradigm which is independent of location. The CVN will be predicted from the preceding state of the vehicle based on position/velocity.

The Society of Automotive Engineers (SAE) has established communication standards for DSRC for connected vehicles (SAE J2735) [38]. This is the first step towards standardizing the CVN communication protocols as most vehicle manufacturers in the near future will be building cars with in-built Wi-Fi capability. An immediate application would be to reduce traffic congestion by relaying an accident/roadworks/incident to re-route traffic thus reducing the overall traffic congestion.

VIII. Conclusions

This paper presents QoS assessment and modelling of CVN. QoS assessment was conducted using ANOVA, PCA and CMD on the Link QoS of connected vehicles. We used data from GEMV[2]. Our analysis shows that for V2V number of transmitting vehicles in the network (neighbours) has a bigger impact than in V2I on link quality. However, parameters of LOS and NLOS are significant in both types (V2V and V2I). This enabled us to model the three parameters of LOS, NLOS and Neighbours on Link QoS and subject it to step response. The step response result shows that the system settles on a steady state. It further addresses QoS challenges in connected vehicles and presents an overview of the various network models and wireless channels and their applications in connected vehicles scenarios. The key issues identified will help lay the foundation for future research directions in this area. Some of the challenges that need to be addressed by wireless channels in connected vehicles are weather conditions and their impacts, for example, how low visibility and extreme weather conditions can impact on the QoS of the connected vehicle. In addition, cameras and ultra-sonic sensors are limited to low distance. The overall reliability of the sensor data within connected vehicle communication is critical. As suggested in [3], for safety management, sensors that can detect fatigue levels of the driver by monitoring various bodily conditions can also be added. The first commercial vehicles to have onboard units installed are expected in summer 2017 from Cadillac [39].

The data information and filters necessary are also investigated, e.g., what is critical, necessary, add-on to process in the vehicle and what data to send/receive to/from the data centre. The challenge is to maintain the QoS of the real-time communication protocol and how to ensure data integrity of the process. With autonomous driving being trialled this year in the UK, what role will connected vehicles play? These are some of the imminent research questions highlighted from our research. Future direction of our research will aim to address the points raised in this paper and focus on refining the modelling of CVN with some form of control.

Acknowledgment

This paper is an invited extended version of the conference paper A. Khan, C. Goh, C-L. Lim and S. Sharma, “Quality of Service Assessment in Connected Vehicles”, published in Proceedings of 6th International Conference on Advances in Vehicular Systems, Technologies and Applications, Nice, France, 23-27 July 2017.

The work reported here is supported in part by the Santander Seed-Corn Research Grant.

References

[1] A. Khan,, C. Goh, C. Lim and S. Sharma, “QoS Assessment in Connected Vehicles”, The 6th Int. Conf. on Advances in Vehicular Systems, Technologies and Applications, Vehicular 2017, 23-27 July, Nice, France, 2017.

[2] “Connected Car Market” by Connectivity Form Factor (Embedded, Tethered, Integrated), Connectivity Technology (LTE, 3G, HSPA+, Wi-Fi, Bluetooth), Application (Navigation, Telematics, Infotainment) and Geography - Trends & Forecast to 2014 – 2020.

[3] IEEE webinar on connected vehicles Part I and II, Dec. 2015. https://www.comsoc.org/blog/live-webinar-bringing-connected-vehicle-technology-next-level-part-2 [Retrieved 20th December 2015]

[4] M. Faezipour, M. Nourani, A. Saeed and S. Addepalli, “Progress and challenges in intelligent vehicle area networks”, Communications of the ACM, vol. 55, no.2, pp. 90-100, Feb. 2012.

[5] N. Lu, N. Cheng, N. Zhang, X. Shen and J. W. Mark, “Connected vehicles: solutions and challenges, IEEE Internet of Things Journal”, vol. 1, no. 4, pp. 289-299, August 2014.

[6] The pathway to driverless cars, summary report and action plan. Department of Transport UK, Feb. 2015.

[7] Y. Fangchun, W. Shangguang, l. Jinglin, L. Zhihan and S. Qibo, "An overview of Internet of Vehicles, Vehicular Networking", China Communication, vol. 11, issue 10, pp. 1-15, Oct 2014.

[8] Y. Kim, Y. Song and J. Lee, "Vehicular authentication security mechanism modelling using petri nets", Indian Journal of Science and Technology, vol. 8(S7), pp. 443-447, April 2015.

[9] P. Diekhake, T. Kurczveil and U. Becker, "Modelling and simulation of distributed time critical communication and control systems in vehicles or vehicle networks", IFAC Conference Proceedings, 49-3, pp 303-308, 2016.

[10] M. D. S. Soares and J. Vrancken, "A modular petri net to modelling and scenario analysis of a network of road traffic signals", Control Engineering Practice, vol. 20, issue 11, pp. 1183-1194, Nov. 2012.

[11] Y. Li, M. Zhao and W. Wang, "Intermittently connected vehicle-to-vehicle networks: Detection and analysis", IEEE Globecom, 5-9 Dec. 2011.

[12] M. Amadeo, C. Campolo and A. Molinaro, "Information-centric networking for connected vehicles: A survey and future perspectives", Automotive Networking and Applications, IEEE Communications magazine, vol. 54, issue 2, pp. 98-104, Feb. 2016.

[13] J. Voas,."Network of 'Things'", NIST Special publication, 800-183, 2016. http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-183.pdf [Retrieved May 2017].

[14] M. Boban, J. Barros and O. Tonguz, "Geometry-based vehicle-to-vehicle channel modeling for large-scale simulation", IEEE Transactions on Vehicular Technology, vol. 63, Issue 9, pp. 4146-4164, 2014.

[15] S. Narla, "The evolution of connected vehicle technology: from smart drivers to smart cars to …. Self-driving cars", ITE Journal, July 2013.

[16] G. Karagiannis, O. Altintas, E. Ekici, G. Heijenk, B. Jarupan, K. Lin, and T. Weil, "Vehicular Networking: A Survey and Tutorial on Requirements, Architectures, Challenges, Standards and Solutions", IEEE Communications Surveys & Tutorials, Vol. 13, No. 4, Fourth Quarter 2011.

[17] S. Céspedes, X. Shen and C. Lazo, "IP Mobility Management for Vehicular Communication Networks: Challenges and Solutions", IEEE Communications Magazine May 2011, pp. 187–194.

[18] G. Samara, W. A.H. Al-Salihy and R. Sures, "Security Issues and Challenges of Vehicular AdHoc Networks (VANET)", , 4th International Conference on New Trends in Information Science and Service Science (NISS), 11-13 May 2010, Gyeongju, South Korea.

[19] F. Dressler, H. Hartenstein O. Altintas, and O. K. Tonguz, "Inter-Vehicle Communication: Quo Vadis", IEEE Communications Magazine, June 2014.

[20] L. Zhang and G. Orosz, "Black-box modelling of connected vehicle networks", In American Control Conference (ACC), Boston Mariott, Copley Place, July 6-8, Boston, MA, 2016.

[21] R. Pandita and D. Caveney, "Preceding vehicle state prediction", IEEE Intelligent Vehicles Symposium (IV), June 23-26, Gold Coast, Australia, 2013.

[22] System Identification toolbox, MATLAB, https://uk.mathworks.com/help/ident/index.html [Retrieved 15th January 2018]

[23] Intelligent Transport Systems (ITS); Communications Architecture, September 2010. ETSI EN 302 665 V1.1.1 (2010-09).

[24] A.Khan, S. Sharma, C. Goh & C.Lim, "A Review of Network Models for Internet of Vehicles", The 6th Int. Conf. on Advances in Vehicular Systems, Technologies and Applications, Vehicular 2017, 23-27 July, Nice, France, 2017.

[25] O. Kaiwartya, A. H. abdullah, Y. Cao, A. Altameem, M. Prasad, C-T. Lin and X. Liu, "Inter of Vehicles: Motivation, layered architecture, network model, challenges and future aspects", IEEE Access, vol. 4, pp. 5356-5373, 28 Sept. 2016.

[26] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. H. Briggs, and R. L. Braynard, "Networking Named Content," in ACM CoNEXT, Rome, Italy, 1-4, Dec. 2009.

[27] S. H Ahmed, S.H. Bouk, M. A. Yaqub, D. Kim, H. Song, and J. Lloret, "CODIE: Controlled data and interest evaluation in vehicular named data networks," IEEE Trans. Veh. Technol., vol. 65, no. 6, pp. 3954-3963, Jun. 2016.

[28] T. T. Gunes, S. T. Kwaong U and H. Afifi, "Hybrid model for LT-assisted D2D communications", Ad-hoc, Mobilr and Wireless Networks conference (AdhocNow), vol. 8487, pp. 100-113, 2014.

[29] M. I. Hassan, H. L. Vu, and T. Sakurai, "Performance analysis of the IEEE 802.11 MAC protocol for DSRC safety applications," IEEE Trans. Veh.Technol., vol. 60, no. 8, pp. 3882–3896, Oct. 2011.

[30] J. Qiongqiong, Y. Zhou, J. Wang, Y. Wang and X. Wang "A lightweight cross-layer cooperative testbed for evaluation of connected vehicles", , 11th International Conference on Mobile Ad-hoc and Sensor Networks (MSN), 16-18 Dec. 2015, Shenzhen, China.

[31] S-H. Yu, O. Shih, and H-M. Tsai, "Smart Automotive Lighting for Vehicle Safety", IEEE Communications Magazine, Dec. 2013.

[32] S. Cespedes and X. Shen, "On achieving seamless IP communications in heterogenous vehicular networks", IEEE Transactions on Intelligent Transportation Systems, vol. 216, No. 6, Dec 2015.

[33] J. B. Kenney, S. Barve and V. Rai, "Comparing communication performance of DSRC OBEs from multiple suppliers", 19th ITS World Congress, 22-26 Oct., Austria, 2012.

[34] J.-M. Girard, N. Tricot, K. Younsi, and J.-C. Popieul, "When does the driver workload reaches its limits?" in Proc. IEEE Intell. Transp. Syst. Conf. (ITSC'06), Toronto, ON, Canada, Sep. 2006, pp. 578–583.

[35] C. Wu and Y. Liu, "Queuing network modeling of driver workload and performance," IEEE Trans. Intell. Transp. Syst., vol. 8, no. 3, pp. 528–537, Sep. 2007.

[36] G. W. Snedecor and W. G. Cochran, "Statistical Methods", 8th ed. Ames:Iowa State Univ. Press, 1989.

[37] S. K. Datta, R. P. F. Costa, J. Ham and C. Bonnet, "Integrating connected vehicles in Internet of Things ecosystems: challenges and solutions", IEEE 17th International Symposium of A World of Wireless, Mobile and Multimedia Networks (WoWMoM), 21-24 June 2016.

[38] Society of Automotive Engineers, "Dedicated Short Range Communications (DSRC) Message Set Dictionary", J2735 Ground Vehicle Standard, J2735, 2016.

[39] General Motors. Cadillac to Introduce Advanced "Intelligent and Connected" Vehicle Technologies on Select 2017 Models. [0nline] [retrieved: May, 2017].

Intelligent Software Development Method Based on Model Driven Architecture

Keinosuke Matsumoto, Kimiaki Nakoshi, and Naoki Mori
Department of Computer Science and Intelligent Systems
Graduate School of Engineering, Osaka Prefecture University
Sakai, Osaka, Japan
email: {matsu, nakoshi, mori}@cs.osakafu-u.ac.jp

***Abstract*—Recently, Model Driven Architecture (MDA) has attracted attention in the field of software development. MDA is a software engineering approach that uses models to create products such as source code. On the other hand, executable Unified Modeling Language (UML) consists of activities, common behavior, and execution models; however, it has not been effectively transformed into source code. This paper proposes a method for transforming executable UML and class diagrams with their associations into source code. Executable UML can describe a system's behavior well enough to be executed; however, it is very difficult for executable UML to handle system data. Therefore, the proposed method uses class diagrams for this purpose. The method can create models independent of platforms, such as programming languages. The proposed method is applied to a system, where Java and C# code was generated from system models, which were generated using an executable UML model; in addition, development costs are evaluated. As a result, it is confirmed that this method can significantly reduce costs when models are reused.**

Keywords—executable UML; activity diagram; model driven architecture; UML.

I. INTRODUCTION

This paper is based on the study presented at INTELLI 2017 [1]. In today's software development environment, software reuse, modification, and migration of existing systems have increased at a greater pace than new development. According to an investigative report [2] by the Information-Technology Promotion Agency (IPA), reuse, modification, and migration of existing systems account for approximately 73.3% of software development and new development accounts for 26.2% as shown in Fig. 1. Many software bugs enter the upper processes, such as requirement specification, system design, and software design. However, bugs are mostly discovered in lower processes, such as the testing process. The need to detect bugs upstream is gaining priority. Under such a situation, software developers require a development technique that is easy to reuse and that adjusts to changes in implementation technique. Model driven architecture (MDA) [3] is attracting attention as an approach that generates source code automatically from models that are not influenced by implementation [4–6]. Its core data are models that serve as design diagrams of software. It includes a transformation to various types of models and automatic source code generation based on the models. Therefore, it can directly link software design and implementation.

The final goal of MDA is to generate automatically executable source code for multiple platforms. For that purpose, it is necessary to make the architecture and behavior of a system independent from platforms, e.g., a Platform Independent Model (PIM) that does not depend on platforms, such as programming languages. Executable Unified Modeling Language (UML) [7][8] is advocated as this type of model as it expresses all actions for every type of processing, and expresses input and output data by a pin in activity diagram, which is one of various UML [9] diagrams. The source code for various platforms can be generated from one MDA-type model because processing and data can be transformed for every platform if executable UML is used.

In this study, a method is proposed that generates source code automatically from executable UML. It is very difficult for executable UML to handle a system's data. To solve this problem, this paper proposes a modeling tool that associates an executable UML with class diagrams and acquires data from them. It can treat not only data, but can introduce the hierarchical structure of class diagrams in executable UML. If the platform of future systems, such a programing language, is changed, software developers cannot reuse existing source code, but they can reuse UML models to automatically generate source code in the new programming language.

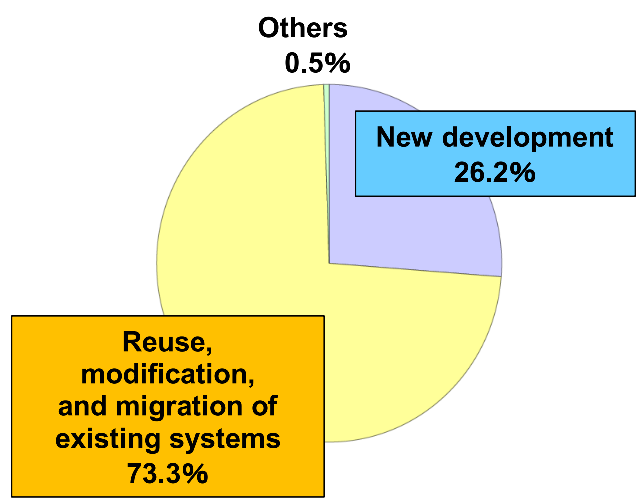


Figure 1. Percentage of software development projects.

The contents of this paper are as follows. In Section II, background of this study is described. In Section III, the proposed method is explained. In Section IV, the results of application experiments confirm the validity of the proposed method. Finally, in Section V, the conclusions and future work are presented.

II. Background

This section describes background of this study that makes use of Acceleo and executable UML.

MDA's core data are models that serve as software design drawings. The models are divided into platform dependent and independent models. Auto source code generation tools (for example, Acceleo [10]) transform models from PIM to a Platform Specific Model (PSM) that depends on a platform, and generates source code automatically. Transformation of the PIM is important and can generate source code in various platforms by applying different transformation rules to each platform. Acceleo is a plug-in of the integrated development environment Eclipse [11], and a code generator that translates MetaObject Facility (MOF) [12] type models into source code on the basis of code transformation rules called a template. Almost all source code generation tools like Acceleo can translate the models directly, but the tools have many constraints. For example, they can only generate skeleton code. In addition, they cannot hold and calculate data. Therefore, they cannot recognize what types of model elements have been read. It is impossible to search the connections between nodes by using graph theory. When branches and loops of activity diagrams are transformed, the generation tools have a problem in that they cannot appropriately transform them because they do not understand the environment.

Executable UML is a model based on activity diagrams, as shown in Fig. 2. It has the following features:

- An action is properly used for every type.
- Input and output data of each action are processed as a pin, and they are clearly separated from the action.
- A model library that describes the fundamental operations in a model is prepared.

Each type of action has respectively proper semantics, and transformation with respect to each action becomes possible by following the semantics. The type and semantics of an action used in executable UML are as follows.

1) ValueSpecificationAction: Outputs a value of primitive type data such as an integer, real number, character string, or logical value.

2) ReadStructuralFeatureAction: Reads certain structural characteristics. For example, it is used when the property of class diagrams is read.

3) ReadSelfAction: Reads itself.

4) CallOperationAction: Calls methods in class diagrams.

5) CallBehaviorAction: Calls behaviors in behavior diagrams.

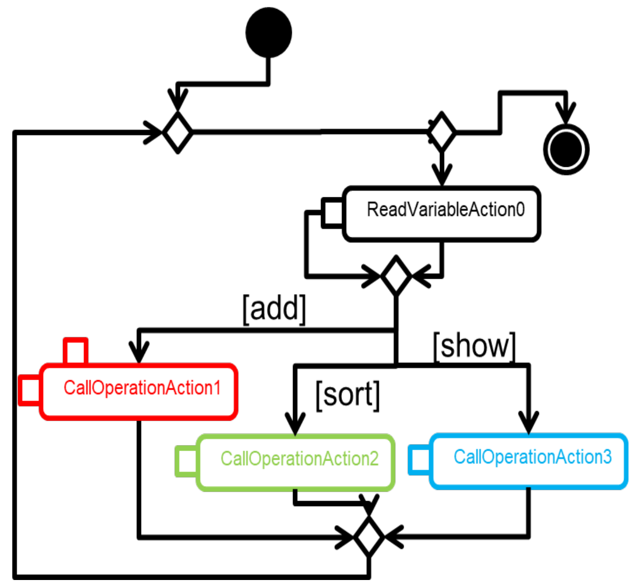


Figure 2. Example of executable UML.

6) AddVariableValueAction: Adds a value to the variable or replaces the variable with its value.

7) ReadVariableAction: Reads a variable or generates one.

8) CreatObjectAction: Creates a new object.

The model library consists of the Foundational Model Library, Collection Classes, and Collection Functions. The contents of the model library are shown below.

1) Foundational Model Library: Offers primitive type data, and their behaviors (four arithmetic operations, comparison, etc.) and all input-output relationships.

2) Collection Classes: Offers the collection classes of Set, Ordered Set, Bag, List, Queue, Dequeue, and Map.

3) Collection Functions: Offers the methods (add, delete, etc.) of the collection class.

The model library is used by calling CallOperationAction or CallBehaviorAction.

III. Proposed Method

This section explains the technique of transforming executable UML to source code. Although executable UML is useful, this model has not been used effectively for automatic generation of source code. Moreover, the handling of data is inadequate if using only executable UML. To solve this problem, a method is proposed for generating source code automatically from executable UML. The method utilizes a modeling tool that associates executable UML with class diagrams. If executable UML requires data, the method retrieves the data from associated class diagrams.

The outline of the proposed method is shown in Fig. 3. Skeleton code is transformed from class diagrams by using Acceleo templates [13] for classes. The skeleton code consists only of class names, fields, and methods that do not have specific values of data. Data and a method to

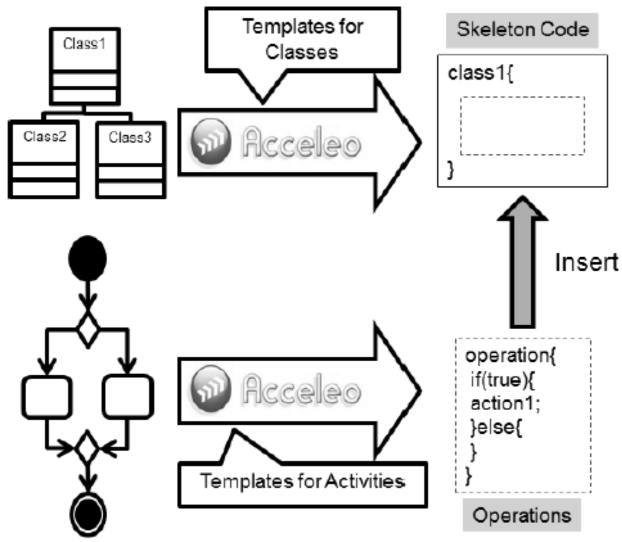


Figure 3. Schematic diagram of the proposed method.

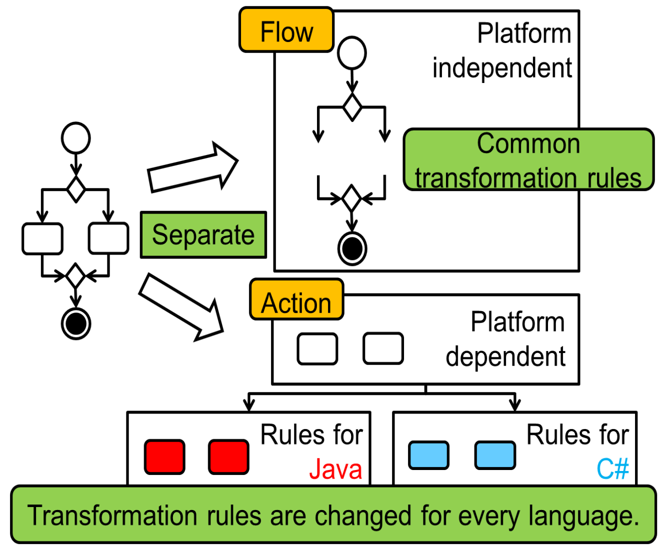


Figure 4. Separation of executable UML.

manipulate the data are automatically generated from executable UML afterward. Since data is associated with class diagrams, other methods and data in the classes are acquirable using this association. Papyrus UML [14] was used for the associations among these models.

A. Transformation from Class Diagrams to Skeleton Code

UML to Java Generator [15] was used to convert the transformation rules from class diagrams to skeleton code. These rules generate the following.

- Connection of inheritance or interface
- Field variables and methods (e.g., getter and setter)
- Names and parameters of member functions

This is a template for Java. When transforming models to C#, several additional changes are required, such as deletion of constructors and addition of ":" for the inheritance relationship.

B. Transformation from Executable UML to Source Code

Executable UML is based on activity diagrams, which consist of actions, data, and their flows. Although transform rules of actions and data differ from platform to platform, the flows are fundamentally the same. Therefore, transformation of flows is separated from transformation of actions and data as shown in Fig. 4. Flows decide the order of transformation of actions and data. This separation can flexibly transform one model to the source code of multiple platforms. The transformation flow of executable UML is described as follows.

1) Transformation of flows

A flow of executable UML is shown by connecting nodes, which include actions and data, with an edge. However, neither a branch nor loop is transformed by only connecting nodes along the flow. On transforming a decision or merge node used for a branch or loop, the proposed method searches a part of the executable UML near the node and provides an appropriate keyword to a connecting node and edge. The method transforms them according to keywords. The keywords given to model elements are shown below.

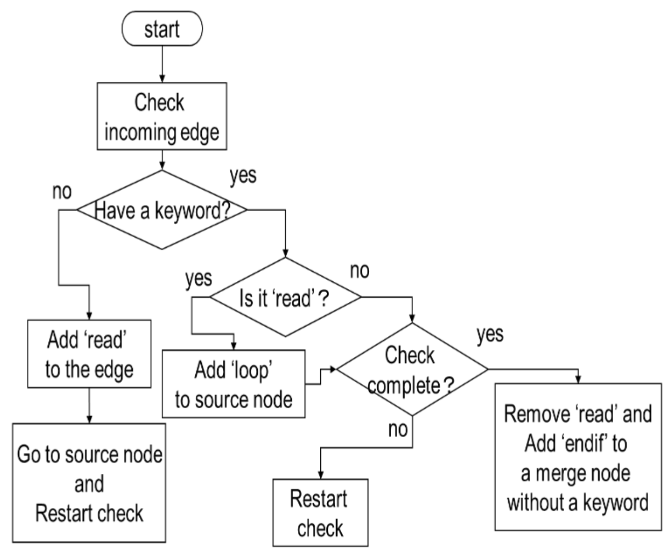


Figure 5. Search algorithm.

a) finish: Indicates a node or edge whose processing has finished.

b) loop: A decision node in the entrance or exit of a loop.

c) endif: A merge node at the end of a branch.

d) read: An edge is being searched.

The flow of the search is shown in Fig. 5 and its algorithm is as follows.

(1) Follow an edge that is not searched in the reverse direction of its arrow.
(2) If an edge and node have not been searched, issue the keyword 'read'.
(3) If a node has the keyword 'read', replace the keyword with 'loop'. If the node is a decision node, assign the keyword 'loop' to a searched edge going out from the decision node.
(4) Repeat the processing of (1) –(3) until there is no edge remaining to be searched.
(5) Remove 'read' after the last edge. If there is a merge node that does not have keyword 'loop', 'endif' will be assigned to it.

2) Transformation of actions and data

Transformation rules of actions and data are prepared for every platform. In executable UML, an action is properly used for every type of processing, and a transformation rule may be defined per action. The flow of transformation processing is as follows.

(1) If a node is an action, it will be transformed and assigned the keyword ‘finish’. Processing will move to the next node. If the action has an input pin, its flow will return and the objects and actions at the starting point of this flow will be transformed.
(2) If a node is a decision node and it has the keyword ‘loop’, it will be transformed by rules for a loop. If the decision node has no keyword, it will be transformed by the rules for a branch. In addition, the nodes and conditional expressions are retrieved from the connecting edges. When ‘finish’ keywords are assigned to these nodes and edges, processing will move on to the next nodes.
(3) If a keyword is assigned to a merge node, it will be transformed according to the rules of the keyword.

Corresponding relationships of actions between Java and C# are shown in Table I. The upper row of ReadStructural FeatureAction in Table I is the case where *<variable>* is specified, and the lower row is the case where it is not specified.

In addition, corresponding relationships of model libraries between Java and C# are shown in Table II. ReadLine and WriteLine are model libraries for input and output. List.size, List.get, and List.add are prepared by Collection Functions, and they are used for output of list capacity, extraction of list elements, and addition of list elements, respectively. Primitive Functions are operations prepared in the primitive type. Collection Functions are used by calling CallOperationAction, and all other functions are used by calling CallBehaviorAction. Variables inputted by pins and operators defined in the library are shown in italics surrounded by < >.

IV. APPLICATION EXPERIMENTS

As an experiment that verifies operation and the development costs of created templates, the system shown below (Figures 6 and 7) was developed using the proposed method. The system receives three commands for adding data to a list, sorting list elements, and outputting data. This system was described by executable UML and class

TABLE I. ACTIONS AND THEIR APPLICATIONS TO EACH LANGUAGE.

Action	Java	C#
CreateObjectAction	new *<object>*	new *<object>*
ReadSelfAction	this	this
ValueSpecification Action	*<value>*	*<value>*
ReadStructural FeatureAction	*<object>*.*<variable>*	*<object>*.*<variable>*
	(*<resultType>*) *<object>*	*<resultType>*.Parse (*<object>*)
CallOperationAction	*<target>*.*<operation>* (*<parameter>*)	*<target>*.*<operation>* (*<parameter>*)
AddVariableValue Action	*<variable>*=*<value>*	*<variable>*=*<value>*

TABLE II. MODEL LIBRARY ELEMENTS AND THEIR APPLICATIONS.

Model Library	Java	C#
ReadLine	(new BufferedReader(new InputStreamReader(System. in))).readLine()	Console.ReadLine()
WriteLine	System.out.println(*value*)	Console.WriteLine(*value*)
List.size	*<target>*.size()	*<target>*.Count
List.get	*<target>*.get(*<index>*)	*<target>*[*<index>*]
List.add	*<target>*.add(*<data>*)	*<target>*.Add(*<data>*)
Primitive Functions	*<x><function><y>*	*<x><function><y>*

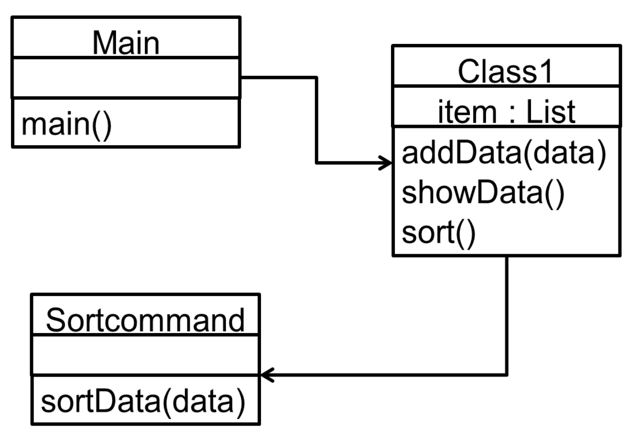


Figure 6. Class diagram of example system.

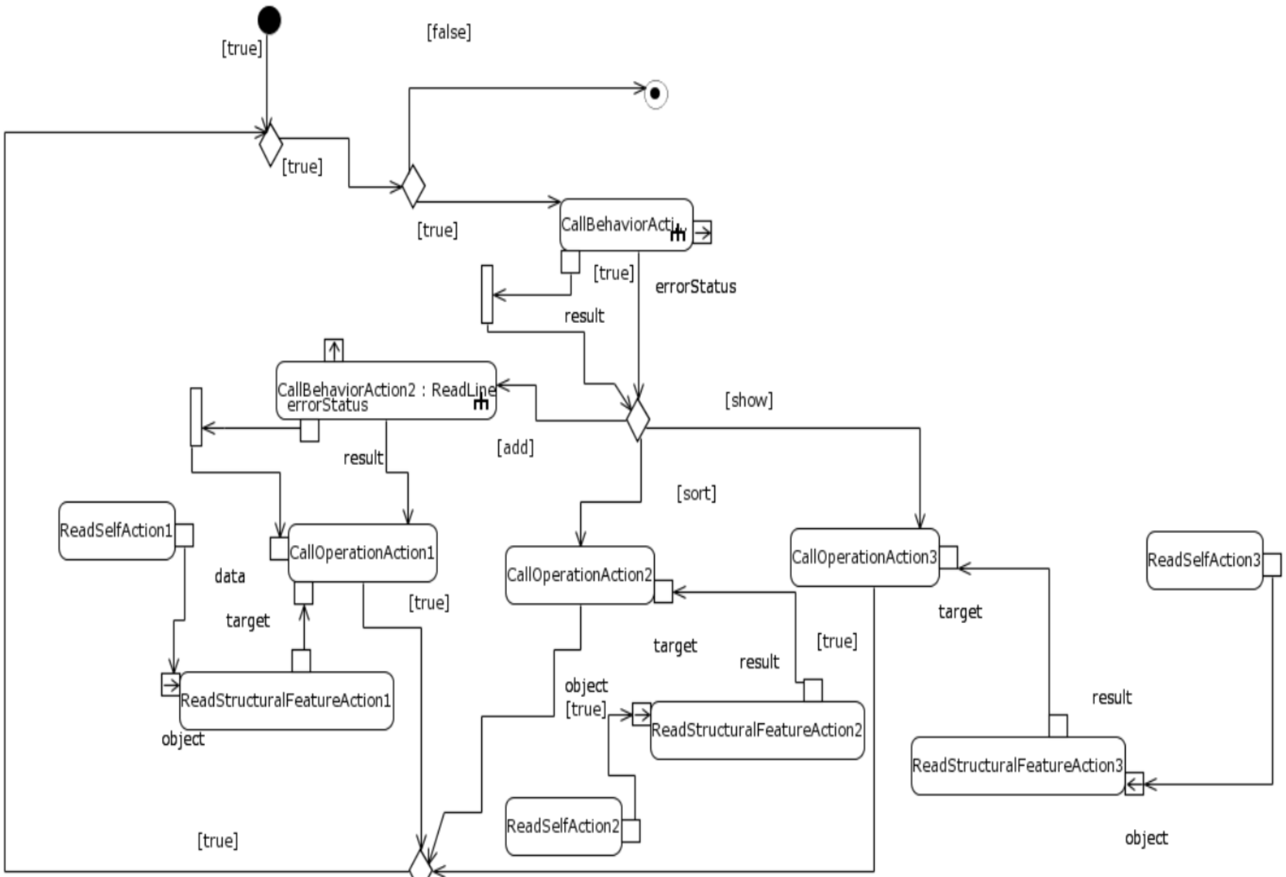


Figure 7. Total system behavior of example system.

```java
package Package1;
import java.util.ArrayList;
public class Class1 {
    private Sortcommand sortcommand=new Sortcommand();
    private List item=new List();
    public Class1() {
        super();
    }
    public void addData(String data) {
        this.item.add(data);
    }
    public void showData() {
        System.out.println(this.item);
    }
    public void sort() {
        this.sortcommand.sortData(this.item);
    }
    public Sortcommand getSortcommand() {
        return this.sortcommand;
    }
    public void setSortcommand(Sortcommand newSortcommand) {
        this.sortcommand = newSortcommand;
    }
    public List getItem() {
        return this.item;
    }
    public void setItem(List newItem) {
        this.item = newItem;
    }
}
```

Figure 8. Generated Java code of Class1.

```csharp
using System;
using System.Collections.Generic;
namespace Package1
{
    public class Class1 {
        private Sortcommand sortcommand=new Sortcommand();
        private List item=new List();
        public  void sort() {
            this.sortcommand.sortData(this.item);
        }
        public  void showData() {
            Console.WriteLine(this.item);
        }
        public  void addData(String data) {
            this.item.Add(data);
        }
        public Sortcommand getSortcommand() {
            return this.sortcommand;
        }
        public void setSortcommand(Sortcommand newSortcommand) {
            this.sortcommand = newSortcommand;
        }
        public List getItem() {
            return this.item;
        }
        public void setItem(List newItem) {
            this.item = newItem;
        }
    }
}
```

Figure 9. Generated C# code of Class1.

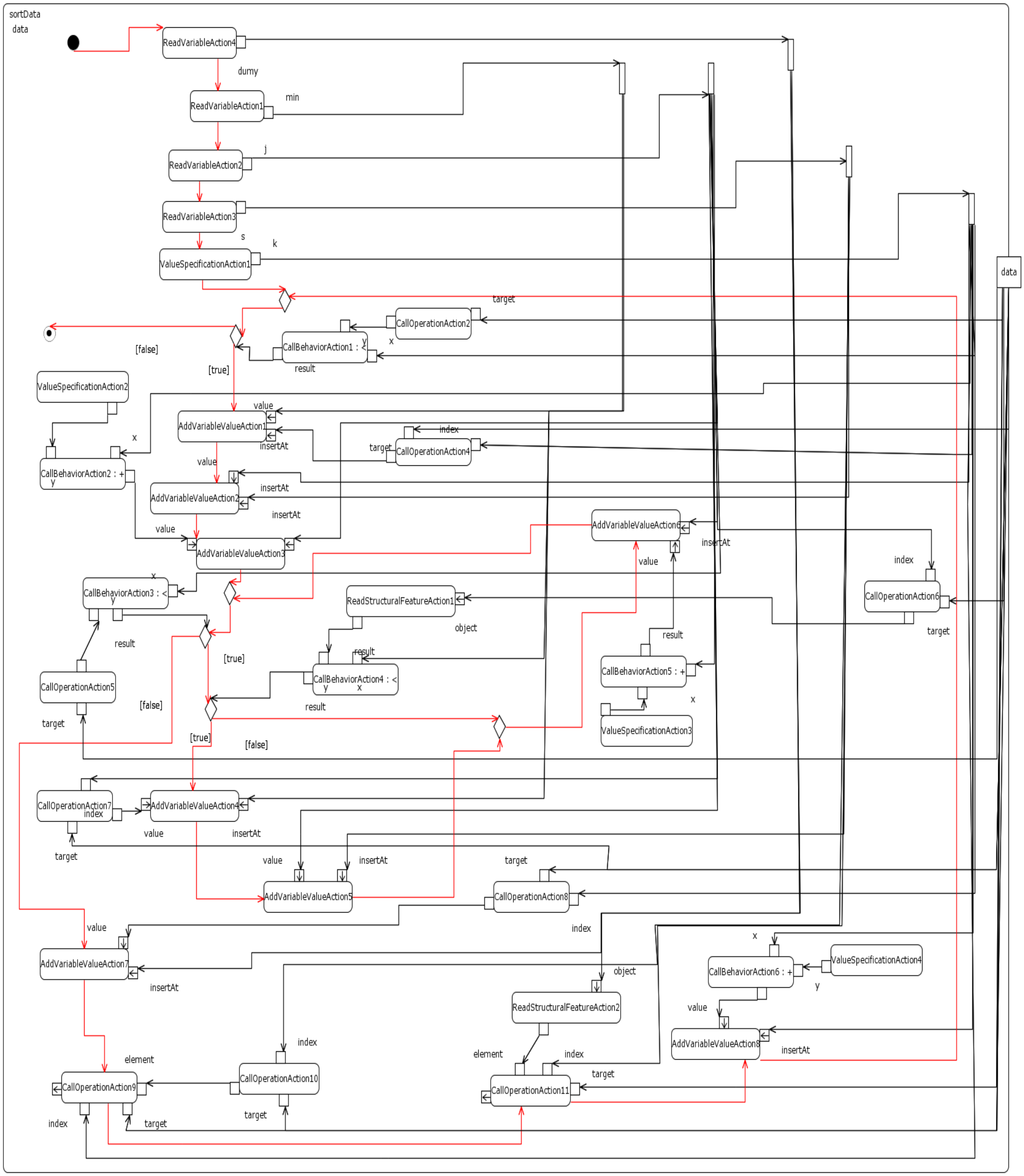


Figure 10. Executable UML of sortData.

```
[template public trs4action(act : Action)]
[if (act.output->notEmpty())]
[let nexN : Sequence(ActivityNode) = act.output.outgoing.target]
[if (nexN->any(oclIsTypeOf(ForkNode))->notEmpty())]
[act.output.type.genType()+' '/][nexN->any(oclIsTypeOf(ForkNode)).name/]=[/if][/let][/if][if (act.oclIsTypeOf(CallBehaviorAction))]
[callBeh(act.oclAsType(CallBehaviorAction))/][elseif (act.oclIsTypeOf(ValueSpecificationAction))]
[valuespe(act.oclAsType(ValueSpecificationAction))/][elseif (act.oclIsKindOf(ReadVariableAction))]
[act.oclAsType(ReadVariableAction).genReadvar()/][elseif (act.oclIsKindOf(CallOperationAction))]
[act.oclAsType(CallOperationAction).callOpe()/][elseif (act.oclIsKindOf(AddVariableValueAction))]
[act.oclAsType(AddVariableValueAction).genAddvar()/][elseif (act.oclIsKindOf(ReadStructuralFeatureAction))]
[act.oclAsType(ReadStructuralFeatureAction).genReadstr()/][elseif(act.oclIsTypeOf(ReadSelfAction))]
[act.oclAsType(ReadSelfAction).genReadSelf()/][else]
[act.name/]([for (it : String | act.input.invalue().genInput())separator (',')][it/][/for])[/if]
[/template]
[template callOpe(ope : CallOperationAction)]
[ope.target.invalue().genInput()/].[ope.operation.genOpe()/]([for (it : String | ope.argument.invalue().genInput())separator (',')][it/][/for])
[/template]
[template public genOpe(ope : Operation)]
[if (ope.class.ancestors(Package).name->includes('Alf'))]
[if (ope.class.ancestors(Package)->reject(oclIsTypeOf(Model)).name->includes('CollectionClasses'))]
[if (ope.name.matches('at'))]
get [elseif(ope.name.matches('replaceAt'))]
set [else]
[ope.name/] [/if] [else]
[ope.name/] [/if] [else]
[ope.name/] [/if]
[/template]

[template public callBeh(act : CallBehaviorAction)]
[if (act.behavior.ancestors(Package).name->includes('FoundationalModelLibrary'))]
[if (act.behavior.ancestors(Package)->reject(oclIsTypeOf(Model)).name->includes('BasicInputOutput'))]
[if (act.behavior.name='WriteLine')]
System.out.println([act.argument.invalue().genInput()/])
[elseif (act.behavior.name='ReadLine')]
[let pin : Pin = act.result->any(oclIsKindOf(Pin))]
(new BufferedReader(new InputStreamReader(System.in))).readLine()[/let][/if][elseif (act.behavior.ancestors(Package)->reject(oclIsTypeOf(Model)).name
[if (act.argument->size()=2)]
[act.argument->any(name='x').invalue().genInput()/][act.behavior.name/][act.argument->any(name='y').invalue().genInput()/][else]
[act.behavior.name/][for (it : String | act.argument.invalue().genInput()) separator (',')][it/][/for][/if][/if][/if]
[/template]
[template public valuespe(act : ValueSpecificationAction)]
[if(act.value.oclIsTypeOf(LiteralString))]
"[act.value.stringValue()/]"[elseif (act.value.oclIsTypeOf(LiteralInteger))]
[act.value.integerValue()/][elseif (act.value.oclIsTypeOf(LiteralBoolean))]
[act.value.booleanValue()/][elseif (act.value.oclIsTypeOf(LiteralReal))]
[act.value.realValue()/][/if]
[/template]
[template public genReadSelf(act : ReadSelfAction)]
this
[/template]
```

Figure 11. A portion of the template for Java.

```
[template public trs4action(act : Action)]
[if (act.output->notEmpty())]
[let nexN : Sequence(ActivityNode) = act.output.outgoing.target]
[if (nexN->any(oclIsTypeOf(ForkNode))->notEmpty())]
[act.output.type.genType()+' '/][nexN->any(oclIsTypeOf(ForkNode)).name/]=[/if][/let][/if][if (act.oclIsTypeOf(CallBehaviorAction))]
[callBeh(act.oclAsType(CallBehaviorAction))/][elseif (act.oclIsTypeOf(ValueSpecificationAction))]
[valuespe(act.oclAsType(ValueSpecificationAction))/][elseif (act.oclIsKindOf(ReadVariableAction))]
[act.oclAsType(ReadVariableAction).genReadvar()/][elseif (act.oclIsKindOf(CallOperationAction))]
[act.oclAsType(CallOperationAction).callOpe()/][elseif (act.oclIsKindOf(AddVariableValueAction))]
[act.oclAsType(AddVariableValueAction).genAddvar()/][elseif (act.oclIsKindOf(ReadStructuralFeatureAction))]
[act.oclAsType(ReadStructuralFeatureAction).genReadstr()/][elseif(act.oclIsTypeOf(ReadSelfAction))]
[act.oclAsType(ReadSelfAction).genReadSelf()/][else]
[act.name/]([for (it : String | act.input.invalue().genInput())separator (',')][it/][/for])[/if]
[/template]
[template callOpe(ope : CallOperationAction)]
[if(ope.operation.class.ancestors(Package).name->includes('CollectionClasses') and ope.operation.name.matches('size'))]
[ope.target.invalue().genInput()/].Count [else]
[ope.target.invalue().genInput()/].[ope.operation.genOpe()/]([for (it : String | ope.argument.invalue().genInput())separator (',')][it/][/for])[/if]
[/template]
[template public genOpe(ope : Operation)]
[if (ope.class.ancestors(Package).name->includes('Alf'))]
[if (ope.class.ancestors(Package)->reject(oclIsTypeOf(Model)).name->includes('CollectionClasses'))]
[if (ope.name.matches('at'))]
[elseif(ope.name.matches('replaceAt'))]
set [elseif(ope.name.matches('add'))]
Add[else]
[ope.name/] [/if] [else]
[ope.name/] [/if] [else]
[ope.name/] [/if]
[/template]
[template public callBeh(act : CallBehaviorAction)]
[if (act.behavior.ancestors(Package).name->includes('FoundationalModelLibrary'))]
[if (act.behavior.ancestors(Package)->reject(oclIsTypeOf(Model)).name->includes('BasicInputOutput'))]
[if (act.behavior.name='WriteLine')]
Console.WriteLine([act.argument.invalue().genInput()/])
[elseif (act.behavior.name='ReadLine')]
[let pin : Pin = act.result->any(oclIsKindOf(Pin))]
Console.ReadLine()[/let][/if][elseif (act.behavior.ancestors(Package)->reject(oclIsTypeOf(Model)).name->includes('PrimitiveBehaviors'))]
[if (act.argument->size()=2)]
[act.argument->any(name='x').invalue().genInput()/][act.behavior.name/][act.argument->any(name='y').invalue().genInput()/][else]
[act.behavior.name/][for (it : String | act.argument.invalue().genInput()) separator (',')][it/][/for][/if][/if][/if]
[/template]
[template public valuespe(act : ValueSpecificationAction)]
[if(act.value.oclIsTypeOf(LiteralString))]
"[act.value.stringValue()/]"[elseif (act.value.oclIsTypeOf(LiteralInteger))]
[act.value.integerValue()/][elseif (act.value.oclIsTypeOf(LiteralBoolean))]
[act.value.booleanValue()/][elseif (act.value.oclIsTypeOf(LiteralReal))]
[act.value.realValue()/][/if]
[/template]
[template public genReadSelf(act : ReadSelfAction)]
this
[/template]
```

Figure 12. A portion of the template for C#.

TABLE III. COMPARISON OF MODEL NODES AND GENERATED LINES.

Number of model nodes	Languages	Number of added or modified lines	Number of finished lines
94	Java	3	74
	C#	5	65

TABLE IV. COMPARISON OF DEVELOPMENT COSTS.

Languages	Production rate	New development cost	Development cost by reusing
Java	96%	131%	4%
C#	92%	152%	8%

diagrams. The source code of Java and C# was generated automatically. The same models as described by the previous processes were used for the model transformation of both languages. Operations were checked by evaluating the source code. Figures 6 and 7 show the class diagram and the behavior model of the system, respectively.

The generated source code in Java and C# for Class1 is shown in Figures 8 and 9, respectively. In addition, Fig. 10 shows the behavior model of Sortcommand, and Figures 11 and 12 show portions of the templates for Java and C#. The generated source code in Java and C# for Sortcommand is shown in Figures 13 and 14, respectively.

The development cost is evaluated according to [16]. It assumes that workload to add one node in UML diagrams equals that to describe one line of source code. Table III shows the number of model nodes, the number of lines of added or modified lines, and finished source code for each language. The added and modified lines correspond to parts that cannot be expressed by executable UML such as package, import, and so on. Table IV shows the rate of automatically generated code to the finished code. In addition, it shows the rate of cost of new development and reuse as compared with manual procedures starting from scratch to completion. The calculation formulas used in Table IV are shown below.

Production rate = (finished code lines – added and modified lines) *100 / finished code lines (1)

New development cost rate = (model nodes + added and modified lines) *100 / finished code lines (2)

Reuse cost rate = (added and modified lines) *100 / finished code lines (3)

Cost of the proposed method is about 130 –160% in developing new software, but it is reduced to less than 10% if reusing the models. According to the investigative report of IPA, ~70% of software development is reuse, modification, and migration of existing systems and new software development is ~30%. If a system is developed by the proposed method, the cost is

$$10*0.7+160*0.3=55\% \quad (4)$$

Although the proposed method is more expensive than manual procedures in new development, it can be less expensive when reusing the model(s) created for a system. Previously-created templates can be used in other projects and the cost declines further by repeating reuse. In the present software development environment, where reuse is common, a large cost reduction can be expected. Systems can be hierarchically divided into several (reusable) classes for every function.

```java
package Package1;
import java.util.List;
public class Sortcommand {
    public Sortcommand() {
        super();
    }
    public void sortData(List<Integer> data) {
        int dumy = 0;
        int min = 0;
        int j = 0;
        int s = 0;
        int k = 0;
        while (k < data.size() == true) {
            min = (int) data.get(k);
            s = k;
            j = k + 1;
            while (j < data.size() == true) {
                if (min < (int) data.get(j) == true) {
                    min = (int) data.get(j);
                    s = j;
                } else
                if (min < (int) data.get(j) == false) {
                }
                j = j + 1;
            }
            dumy = (int) data.get(k);
            data.set(k, data.get(s));
            data.set(s,dumy);
            k = k + 1;
        }
    }
}
```

Figure 13. Generated Java code of Sortcommand.

```
using System;
using System.Collections.Generic;
namespace Package1
{
public class Sortcommand {
    public void sortData(List data) {
        int dumy=0;
        int min=0;
        int j=0;
        int s=0;
        int k=0;
        while(k<data.Count==true){
        min=data[k];
        s=k;
        j=k+1;
        while(j<data.Count==true){
        if(min<(data[j])==true){
        min=data[j];
        s=j;
        }else
        if(min<(data[j])==false){
        }
        j=j+1;
        }
        dumy=data[k];
        data.set(k,data[s]);
        data.set(s,dumy);
        k=k+1;
        }
    }
  }
}
```

Figure 14. Generated C# code of Sortcommand.

V. CONCLUSION

Based on the trend where the rate of reuse, modification, and migration of existing systems is increasing in software development, an MDA method that uses executable UML jointly with class diagrams was proposed in this paper. The key idea and objective of the proposed method are to automatically generate source code that skeleton code does not have. As the result, the proposed method associates class operations with executable UML. Source code in Java and C# was generated from system models, and development costs were evaluated.

If the platform of a system is changed in the future, software developers cannot reuse existing source code, but they can reuse UML models to automatically generate source code in the new programming language. As a result, the proposed method can significantly reduce costs when models are reused. The proposed method can transform models into source code written in any type of programming language if there is an appropriate template. However, the method cannot correspond to a large scale of activity diagrams that contain a lot of classes and methods.

As future work, we believe it will be necessary to decide on a standard of model partitioning and a notation system for objects. In addition, an important future task will be to investigate what types of problems will occur when models are changed.

ACKNOWLEDGMENT

This work was supported in part by JSPS KAKENHI Grant Number JP16K06424.

REFERENCES

[1] K. Matsumoto, K. Nakoshi, and N. Mori, "Intelligent software development method by model driven architecture," Proc. of the Sixth International Conference on Intelligent Systems and Applications (INTELLI 2017), IARIA, July 2017, pp. 7-12, Nice, France.

[2] Information-Technology Promotion Agency, "Actual condition survey on software industry in the 2011 fiscal year," (in Japanese) [Online]. Available from: https://www.ipa.go.jp/files/000004629.pdf, 2018.05.22.

[3] S. J. Mellor, K. Scott, A. Uhl, and D. Weise, MDA distilled: Principle of model driven architecture. Addison-Wesley Longman Publishing Co., Inc. Redwood City, CA, 2004.

[4] T. Buchmann and A. Rimer, "Unifying modeling and programming with ALF," Proc. of the Second International Conference on Advances and Trends in Software Engineering, pp. 10-15, 2016.

[5] M Usman, N. Aamer, and T. H. Kim, "UJECTOR: A tool for executable code generation from UML models. In Advanced Software Engineering and Its Applications, ASEA 2008, pp. 165-170, 2008.

[6] Papyrus User Guide, [Online]. Available from: http://wiki.eclipse.org/Papyrus_User_Guide, 2018.05.22.

[7] Object Management Group, "Semantics of a foundational subset for executable UML models," [Online]. Available from: http://www.omg.org/spec/FUML/1.1/, 2018.05.22.

[8] Object Management Group, "List of executable UML tools," [Online]. Available from: http://modeling-languages.com/list-of-executable-uml-tools/, 2018.05.22.

[9] Object Management Group, "Unified modeling language superstructure specification V2.1.2," 2007.

[10] Acceleo: [Online]. Available from: http://www.eclipse.org/acceleo/, 2018.05.22.

[11] F. Budinsky, Eclipse modeling framework: A developer's guide. Addison-Wesley Professional, 2004.

[12] Object Management Group, "Metaobject facility," [Online]. Available from: http://www.omg.org/mof/, 2018.05.22.

[13] Acceleo template: [Online]. Available from: https://wiki.eclipse.org/Acceleo/User_Guide#Extract_as_Template, 2018.05.22.

[14] A. Lanusse et al. "Papyrus UML: An open source toolset for MDA," Proc. of the Fifth European Conference on Model-Driven Architecture Foundations and Applications, pp. 1-4, 2009.

[15] UML to Java Generator [Online]. Available from: https://marketplace.eclipse.org/content/uml-java-generator, 2018.05. 22.

[16] K. Matsumoto, T. Maruo, M. Murakami, and N. Mori, "A graphical development method for multiagent simulators," Modeling, Simulation and Optimization - Focus on Applications, Shkelzen Cakaj, Eds., pp. 147-157, INTECH, 2010.

A Process-Oriented Evaluation Framework for In-Memory Technology Applications

Stephan Ulbricht*, Marek Opuszko*, Johannes Ruhland*
*Friedrich Schiller University Jena
Department of Business Informations, Jena, Germany
Email: stephan.ulbricht@uni-jena.de, marek.opuszko@uni-jena.de, johannes.ruhland@uni-jena.de

***Abstract*—For several years, it has been predicted that In-memory based IT-systems will become a key technology in addressing the challenges of increasing data volumes and computational speed requirements. Nevertheless, many companies seem reluctant to switch to this technology. Missing application scenarios are often cited as the main cause of the slow spreading. To adress this research gap, this work will introduce a framework for the analysis and evaluation of potential applications of In-memory IT-Systems. The developed framework offers researchers as well as the corporate sector the opportunity to evaluate the suitability of such IT systems for existing as well as future use cases. The development process follows the approach of the design science research. In the first phase relevant influencing factors for the use of In-memory systems are identified conducting a literature review. In the second phase an expert survey is carried out for the evaluation and the identification of further influencing factors. The results show that an acceleration of IT processing does not generate substantial added value. In particular, business-related factors have been neglected in the past. Therefore, a structured analysis framework is introduced considering both, data and analysis factors as well as economical factors. In the last step the introduced framework is applied based on selected cross-industry uses cases to underline the evaluation capabilities.**

Keywords–In-Memory IT-Systems; Case Study; In-Memory Computing; In-Memory Database.

I. INTRODUCTION

In this work, we introduce a design science based system, able to identify and evaluate influential factors for potential application scenarios of In-memory IT-systems [1]. The aim of this approach is to examine existing as well as potential future scenarios. Based on an analysis framework, the requirements and their feasibility of use cases are examined. In order to identify possible influence factors of In-memory application scenarios, case studies and scientific literature are analyzed. Subsequently, the influence factors found are evaluated with the help of field experts who participated in an expert survey, also identifying yet unknown and additional factors.

In December 2014, Amazon introduced the ”Prime Now” service, which guarantees the delivery of several thousands of products within an hour [2]. In the field of high frequency trading, fractions of a second can determine profit or loss [3]. Sociologists have been talking about this subject as the ”age of acceleration” for quite some time [4]. Never before in history were decision makers forced to make entrepreneurial decisions under greater time pressure than today. Furthermore, increasingly huge and heterogeneous data sets are challenging companies. Due to the increasing computational demands, conventional IT solutions reach their performance limits more and more frequently. One of the most promising technologies for solving these challenges are In-memory-based IT systems (IMIS). Although the technology was subject to high expectations in the past, the predicted boom has not yet begun. In this context, many companies complain about the lack of useful and economical application scenarios [5][6]. In a study by the American SAP user group, this point is mentioned as one of the main causes for the delayed distribution [7]. The reasons for this is, among others, the previous focus on technical aspects [8]. A study by the market research company PAC [9], on the other hand, shows that the In-memory technology is of great interest to many companies and can play an important role in the future. 36% of the surveyed company representatives see this technology as an important building block in future IT landscapes. In this field of tension, it becomes clear that the In-memory technology has great potential that has not yet been exploited. Our contribution starts at this point. With the aid of our framework practitioners are able to assess the potential of IMIS.

The paper is organized as follows. Section II introduces the technical background and the related work in the field of IMIS. In Section III the research methodology is presented. Afterwards the results of the literature review and the expert survey are presented in Section III. Section IV comprises the conception and structuring of the framework. The application of the developed framework is shown based on selected use cases in Section V. The final section summarizes the contributions to practice and research.

II. RESEARCH BACKGROUND

The idea of using main memory to store data is not new. These concepts were introduced in the 1980s and 1990s [10][11]. At that time, the main focus was a very fast response times which were realized by main memory databases. The speed advantage in comparison to conventional hard disk data access is illustrated in Table I. Due to high costs and low memory sizes, the interest regarding In-memory databases decreased and the technology almost fell into oblivion. With the introduction of the HANA platform [12], the IT-software provider SAP has once again placed the focus on IMIS.

TABLE I. Data Access Overview (cited from [13]).

Action	Time
Main memory access	100ns
Read 1MB sequentially from memory	250 000ns
Disk seek	5 000 000ns
Read 1MB sequentially from disk	30 000 000ns

A. Problem Context

The previous concerns about the durability of the stored data could be eliminated by the use of non-volatile RAM [14]. The concept of IMIS includes more than a pure data storage in the main memory. In contrast to conventional relational databases, the data is no longer stored row-based, but most often column-based [15]. The concept of a column-based data storage is illustrated in Figure 1. The advantage of a column-based storage is on the one side a better data compression and on the other side a better suitability for analytical tasks.

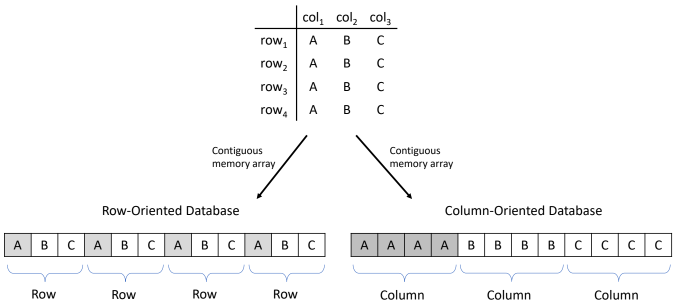


Figure 1. Row- and Column-oriented data layout (adapted according to [13])

Originally, the main application area of IMIS were fast and flexible analysis of large amounts of data in data warehouses. In the meantime, the application areas were extended to transaction systems. The goal here is to dissolve the historically grown separation between online analytical processing (OLAP) and online transaction processing (OLTP) systems [16][17][18]. These hybrid systems are referred to as Online Mixed Workload Processing (OLXP) [19] and Hybrid Transactional/Analytical Processing (HTAP) [20].

The advantage of a common data storage is the elimination of extract, transform, load (ETL) processes from the OLTP into the OLAP system. In addition, transactional data can be used for analytical and planning tasks. Furthermore, there is a potential for savings through the elimination of an additional system [13]. However, it is important to note that analysis and transaction systems have fundamentally different characteristics and requirements [21]. Analytical systems are generally used for the support of specialists and executives. Decisions at these company levels are, in most cases, characterized as strategically or tactically, that means for a longer period. The data access during the execution of analysis are almost exclusively read-only [21][22]. On the other hand, transaction systems are used to solve everyday business tasks of a company. In most cases, the time horizon only covers a relatively short period [23]. The typical transactional workload is also largely read access, but compared with analyzes, with a significantly higher proportion of write accesses [21]. The merging of OLAP and OLTP systems to an OLXP / HTAP system leads not only to the advantages mentioned, but also to problems and difficulties. From a technical point of view, hybrid workloads (line / column-based & read / write) must be simultaneously processed [24][25][26]. The merging to a common information system also leads to a stronger dependency on the respective system provider. In order to be able to exploit the entire benefit of an IMIS, a large number of applications and processes have to be adapted.

B. Related Work

IT providers, such as SAP have predominantly driven the hype surrounding the In-memory technology in the past years. The focus of recent developments was mostly technology-oriented. Similar tendencies can be found in early scientific contributions. Mainly technical features, such as the column-based storage of data [15], data compression [27] or the persistence of volatile storage media [28] were investigated. An alternative approach for the analysis of possible In-memory applications tries to assess the advantages and potentials on the basis of business requirements. In the first publications in this area [29][30][31] Piller and Hagedorn are investigating factors for evaluating In-memory applications. The authors examine the potential of IMIS in the retail sector. Despite the early stage of this technology at the time of the investigation, initial application patterns have already been identified. Similar results are also reported by Cundius et al. in their work [32][33]. They developed a model for evaluating real-time IT systems. The focus of this work was on the workflow-specific properties of real-time IT systems. In [34] the characteristics of In-memory systems were described. Based on the identified attributes application capabilities were derived. As mentioned in the previous section, the starting point of the IMIS developments is the acceleration of analytical applications. In the field of real-time analytics Nadj and Schieder [35] evolved a taxonomy for the characterization of real-time business intelligence.

The use of IMIS not only has an impact on data processing, but also on the downstream decision-making and implementation processes. Vom Brocke et al. examine the connection between the In-memory technology and the resulting business use in their contributions [36][37][38]. They conclude that a value-creation for companies is strongly related to the adaption of processes. Vom Brocke et al. as well as Bärenfänger et al. [39] conclude that the introduction of In-memory technology not only leads to a direct benefit, but to a large extent to downstream improvements in the process flow. Meier et al. further pursue the aim of an economic evaluation in [40]. They also divided the economic effects into direct and indirect attributable effects.

One of the most important innovations of IMIS is the combination of analysis and transaction systems. Winter et al. analyze the properties of IMIS in one of the first case studies [41]. In addition to the volume of data, the integration of the analysis and transaction system is identified as the most important indicator for the assessment of IMIS. This point is also highlighted in several other scientific papers in this field [19][29][30][42]. From a solely technical perspective, IMIS offers huge potential. However, the question arises which companies or application areas can exploit this potential in practice. For many companies predefined reports and evaluations on a daily basis will still be sufficient. For others, the use of real-time data can become a decisive competitive advantage.

Another circumstance that influences the valuation and therefore also the decision concerning the use of IMIS is the uncertainty about the possible performance benefits. Research in this area has shown that In-memory databases do not always perform better than traditional relational databases. A prerequisite for a real performance advantage, for example, is a certain amount of data volume and the number of users [43].

Previous application examples often refer to very specific or exotic tasks. A popular example of the application of IMIS is the analysis of sports data, e.g., in Formula 1 [44] or soccer [45]. Although these examples are quite illustrative, they are not suitable to provide insights into the solution of ”everyday” business problems. The lack of economic use cases is regarded as one of the main obstacles to the adoption of IMIS. This is mentioned in scientific literature [29][42][46] as well as from a company point of view [5][8].

III. Research Methodolgy

The goal of this research work is to examine and structure IMIS use cases with regard to their success factors. The methodology of Design Science Research [47] was used to develop the framework. The aim of this approach is to make comprehensible, scientifically sound and practice-relevant statements in the context of information systems. The design process is not static, it allows changes to be incorporated into the existing model. In the so-called design science research cycle, the results repeatedly iterate through the various stages of the model development [48]. The whole design science research cycle as adapted from [48] is shown in Figure 2. The approach comprised three pillars, evironment, design science research core and knowledge base. The environment pillar is where the object of interest and the application domain reside [47]. This includes organizations, people and technologies. The knowledge base is comprised by foundations and methodologies. Hevner et. al [47] state that this includes foundational theories, frameworks, instruments, constructs, models and methods. The approach further contains three cycles within and between the pillars to ensure that both the relevance and a rigor methodology are ensured during the design phase. The advantage is that the dynamic characteristics of business needs can be directly taken into account. The detailed elaboration of the three phases is described in the following section:

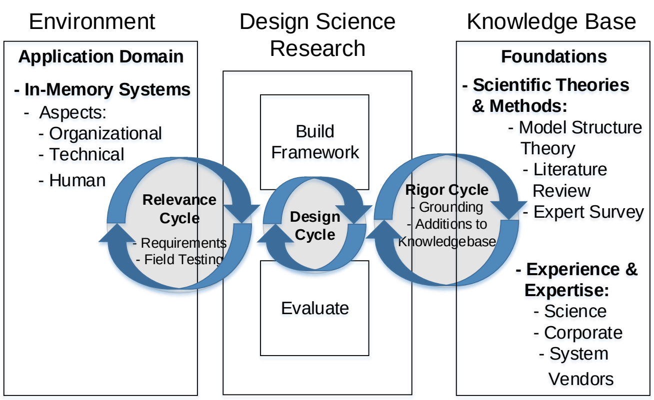


Figure 2. Design Science Research Cycle (adapted according to [48])

Relevance Cycle: The goal of each research is the development of an idea or model which is relevant for the application domain and improves its environment. To reach this goal we validated the relevance of our model multiple times. Starting point and motivation of this research project was the already mentioned demand from business as well as science. During the development the relevance was ensured by the inclusion of experts from science, companies and system providers.

Rigor Cycle: Performing scientific work requires the consideration of thoroughly and established methods with regard to the problem context. To address these requirements we have evaluated and selected appropriate methods for the construction of our framework. The methodology described by Klein and Scholl [49] was used to define the overall structure of the framework. The main advantage of the used methodology is the avoidance of structural defects during the modeling phase. Hereby, it was possible to develop a well-designed and feasible decision model. For this purpose, the scope of the model was first restricted in order to consider only the aspects, which are relevant for the problem solving. After the relevant influential factors were identified, they were subdivided through a structural analysis. As a result of this structuring process, an operationalizable target system for assessing and analyzing In-memory use cases has been created.

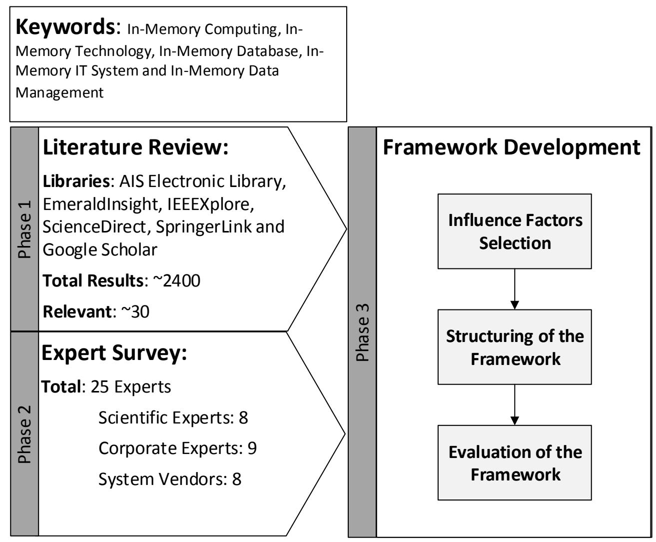


Figure 3. Illustration of the research methodology

Design Cycle: In order to gather the basic factors influencing the framework, scientific work and previous case studies in the field of IMIS were analyzed and evaluated during the first design phase. In terms of research method, this was accomplished according to Webster and Watson [51]. In the literature review, established literature databases (AIS Electronic Library, EmeraldInsight, IEEEXplore, ScienceDirect, SpringerLink and Google Scholar) were investigated. The search included the following key words: "In-Memory Computing", "In-Memory Technology", "In-Memory Database", "In-Memory IT System" and "In-Memory Data Management". Subsequently, a backward search was carried out. Therefore, only papers dealing with the application and the business perspective of IMIS were used. The study of the literature databases revealed that around 2400 scientific publications have so far dealt with IMIS. During the litertaure review phase we conducted a backward as well as a forward search as suggested by Webster and Watson [51]. In the backward search, the quoted sources of the keyword search were analyzed to determine the results of prior research. Based on these results we used Google Scholar to identify the citing articles. Due to the context of this paper only publications with an business perspective were considered. Hence, 30 relevant papers remained. The detailed results are explained in the next section. During the second design phase a qualitative expert survey [52] was carried out to evaluate the results and identify further factors. In particular, the expert survey was carried out to reveal further findings on challenges from an economic point of view. In order to cover a broad range of opinions and experiences, the experts were composed of representatives from different fields. In total 25 experts in the field of IMIS were interviewed. These included scientists, company representatives as well as representatives of leading

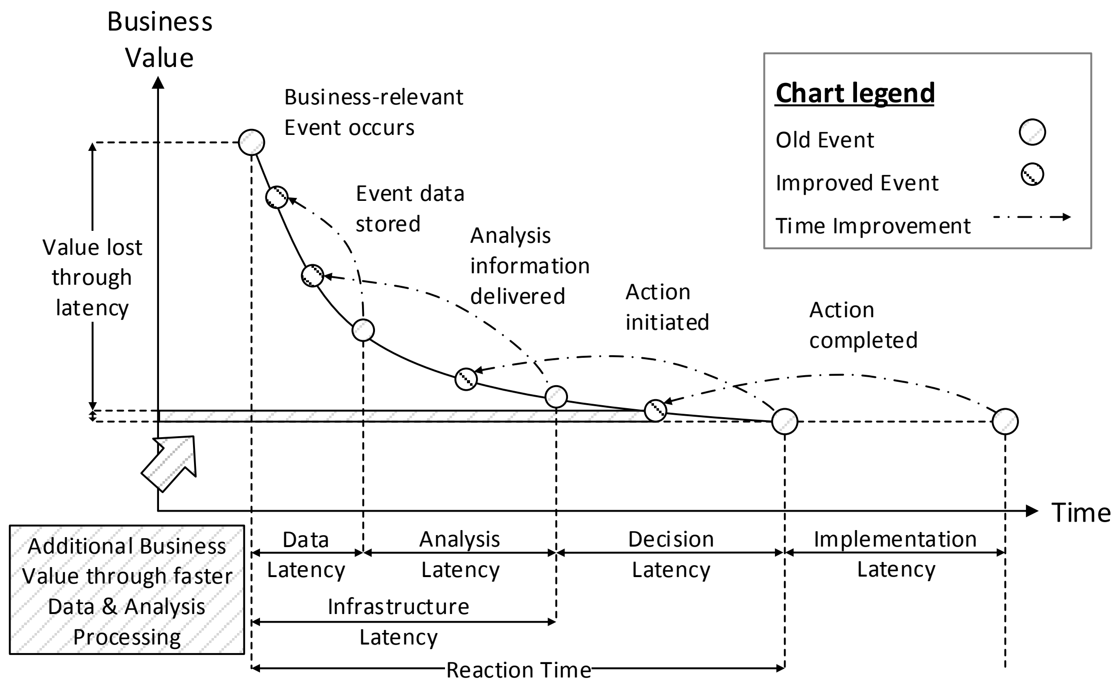


Figure 4. Correlation between time and business value (adapted according to [50])

IMIS providers. To ensure a comprehensive knowledge base we only included experts with a proven experience with In-memory systems. The scientific experts were selected based on publications in the area of IMIS as well as knowledge in the field of databases and information systems. The interviewed representatives of the system providers and the companies were also selected on the basis of their experience in this area. Because of the relative high aquistion costs of IMIS systems only corporate representatives from medium and large companies have been included. In semi-structured interviews the experts were asked about the potential and the obstacles of the In-memory technology. In addition, the experts were asked to evaluate possible application scenarios and their characteristics in detail. An overview of the research methodologies is shown in Figure 3.

IV. Results from the Literature Review and Expert Survey

This section outlines the steps of the design cycle process. The results of the literature review are presented in the first part. The findings from this part form the foundation of the created artifact. In the second part, the results of the expert survey are presented. In the context of the design science cycle, the expert survey is used to evaluate the previous results as well as to reveal new influence factors.

A. Results of the literature and case study review

The examined contributions used different approaches to deal with the analysis and assessment of the scenarios. The work [29] by Piller and Hagedorn has proven to be a suitable basis for the model presented in this work. Starting from the business process characteristics described in this study, further influencing factors were identified and classified.

Main memory-based databases are often mentioned to solve the challenges which are associated with so called Big Data applications. Due to the availability of larger main memory and advanced compression by the column orientation, IMIS is able to process large amounts of data [13][53]. Therefore, it is appropriate to include the data volume of a use case into the consideration. Apart from the data volume, a number of other factors play a decisive role. These include, for example, the urgency of the results [29][30][38][50] or the dynamics of the data [29][30][32]. Hence, high-performance systems have a strong positive effect if the data changes frequently. If the underlying data changes only rarely and to a small extent, the potential additional value of a real-time result is very limited. An example for this are purchase proposals in large online shops based on customer segmentation, which change in general only rarely or marginally. A further influencing parameter is the number and type of source systems [54][55]. In order to cover a broad range of information, it may be advantageous to integrate several different source systems. However, from a critical point of view, problems emerge. The transmission from external sources can lead to delays. A further and currently very often-discussed topic is the veracity of information [56].

As already mentioned in Section II, business processes must be adapted with regard to the newly gained flexibility and speed of data analysis in order to exploit the full potential [32][36][37]. The need for process adaptation has to be clarified on the base of the time business-value relationship concept from Hackathorn [50]. Figure 4 visualizes this concept and shows that the information-processing latency caused by IMIS can be reduced, but the additional business profit is relatively low. In order to generate a higher added value, it is also necessary to modify and accelerate the downstream decision-making and implementation processes.

B. Results of the expert survey

In order to evaluate the results and identify further influential factors, a semistructured expert study was conducted. One of the most frequent points mentioned in the interviews was the high uncertainty regarding the economic benefit. Despite the decline in hardware costs, the purchase of a main memory-based information system is associated with both high investment costs and a significant total cost of ownership [40]. As with any other investment decision, sufficient value must be generated to cover the cost of acquisition. A large proportion of the interviewed company representatives have criticized the poor cost-benefit ratio concerning IMIS and mentioned several causes. In most business applications, mainly "conventional" analysis and evaluations are carried out in the information system. These are already defined in advance or can be well predicted and scheduled. Due to the tactical or strategic character of the decisions, there is no exceptional urgency to obtain the results in most cases.

Out of traditional OLAP tasks, the In-memory technology is perceived more positively. This includes, for example, the areas of predictive maintenance or the integration and analysis of social media. To implement a predictive maintenance, a large number of sensors must be integrated into the analytical system. The continuous measurement results in a high volume of data. Ideally, these data should be analysed as quickly as possible. Another example is the processing of social media, where large quantities of unstructured texts have to be processed. These two examples already confirm a significant proportion of the influencing factors from the first design phase. Another important criterion frequently mentioned were implementation conditions. According to the experts, not only the speed of decision-making is a relevant factor, but also the technical effort and legal obstacles that have to be considered. These factors were not taken into account in the previous literature. Efforts for the indoor localization or digital price tags were cited as examples for technical obstacles. An example for legal obstacles are the data privacy laws regarding the analysis of personal data, especially in european countries like Germany, Spain or the Netherlands.

V. Concept and Structure of the Evaluation Framework

The literature review as well as the results from the expert study make clear that a variety of factors influence the assessment of IMIS scenarios. To support the IMIS decision process it is necessary to structure the influence factors. According to Klein and Scholl [49] the decision factors were divided into a goal hierarchy. This enables a better unterstanding and usability of the model. Based on the results of the literature review, the factors can be clustered into two main categories: data and analysis factors. In the category analysis factors, a large part of the investigations dealt with questions of urgency, complexity or flexibility of analysis. Another segment of research focuses on data-driven factors. These include, among other issues, the volume, the topicality and the dynamics of data. As the hesitant spread of IMIS shows, the technical advantages alone are not enough to generate a substantial benefit. In the past research of IMIS, this fact was rarely taken into account. To consider aspects which are related to, e.g., real-time decisions and to take the results of the expert study into account, the framework was extended by the category of economic factors. This category contains factors with regard to internal as well as external implementation conditions, which are particularly important in the corporate context. This part of the goal hierarchy comprises, e.g., the legal framework or the target group acceptance. The resulting decision model is summarized in Figure 5.

Another issue to be considered is the distinct impact of the influence factors. The different characteristics of the factors show that some have a positive effect on the use of IMIS, while others have a negative impact. To take this into account, we have extended the IMIS evaluation framework by an additional influence indicator. The positive influences are quite obvious on the basis of the technical characteristics. In case of, e.g., a high urgency or a strong data dynamic, the In-memory technology can point out its advantages. An example for factors which are limitating the benefit of IMIS is the amount and kind of source systems. The integration of numerous external source systems can lead to latencies regarding the data provision. Another example are limitations caused by legal regulations. In several countries privacy policies prohibit the analysis of personal data. For a better understanding the decision criterias of our framework are explained in the following part:

A. Analysis factors

Urgency: How fast are the results of the evaluations needed? These factor focuses on the required response time for results of the IT-system, for instance, if deadlines have to be met or subsequent process depend on the results of upstream processes along a critical path. Another example are ad hoc reports requested by high management level [29][30][36][53].

Complexity of the evaluations: How complex is the calculation of the results? The complexity of the calculation depends on the algorithms used and the underlying data [19][29][30].

Flexibility of analysis: Is it more common for new analysis to be performed spontaneously? So far, reports have often been pre-aggregated. The creation of spontaneous evaluations may take a significant amount of time [19][29][41][66].

Degree of detail of the evaluations: In previous information systems, the analyzed data is mostly pre-aggregated. With the help of modern IT systems, evaluations of every level of detail can be implemented [29][46][55].

Integration of Analysis and Transaction System - hybrid workload: This involves the processing of hybrid workloads, here both transactional and analytical data are used for the tasks to be accomplished. These hybrid systems are called OLXP or HTAP. The advantage of shared data management is the elimination of extraction, transformation and loading (ETL) processes from the OLTP system to the OLAP system. In addition, real-time data can be used for analyzes, evaluations or planning [19][29][30][41][42].

Number and type of data sources: From how many sources and from which types the data is obtained. Internal sources are easier to implement and faster to retrieve in most

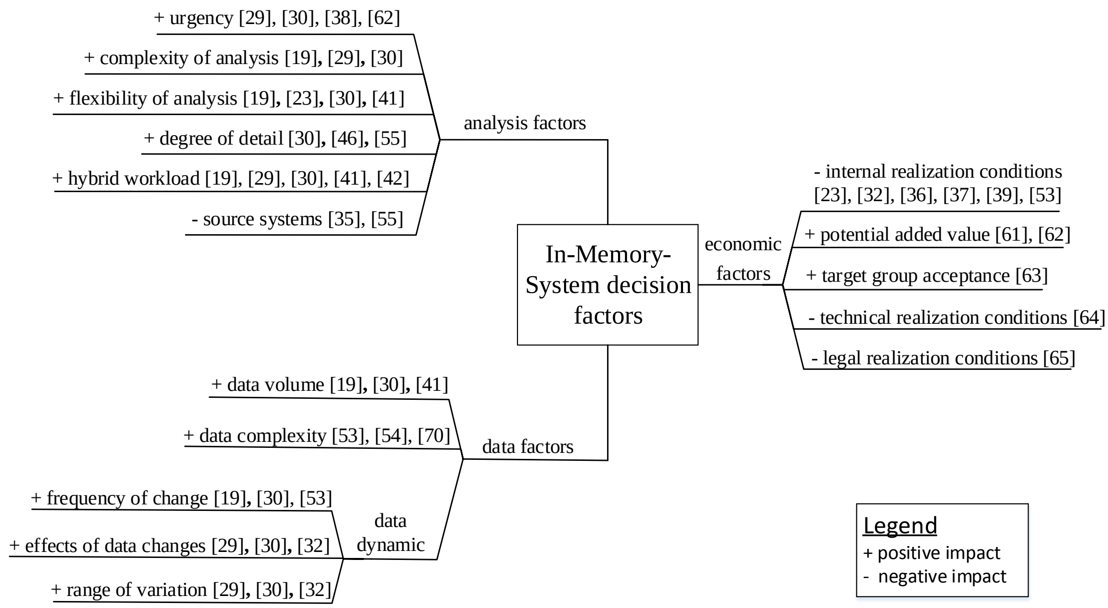


Figure 5. Overview of the analysis and evaluation framework

cases. They further offer a higher predictability for changes in the interface, e.g., changes in the API. For external sources, such as social media or cloud services, however, delays may occur and interfaces have to be updated on a regular basis [55][54].

B. Characteristics of the underlying data - data factors

Data volume: How big is the underlying dataset. In general the use of In-memory systems is economically more reasonable with large data volumes. Smaller amounts of data can be usually processed by conventional means [19][29][41].

Data Complexity: In addition to volume and the rate of change, the complexity of data plays a crucial role in big data. The complexity depends on the one hand on the format and on the other hand on the data structure. Data can be categorised as structured in a table like manner, semi-structured like XML, JSON, text files or unstructered data, such as images and video. Especially the processing of unstructered data usually requires additional methods, such as text-mining and filtering and enhancing images in order to perform image recognition techniques [57][58][60].

Characteristics of the underlying data - data dynamics

Frequency of change - topicality of the data: Which requirements are made in terms of timeliness? Decisive here is how much added value can be generated by the second-precise data in comparison to data that is updated in a minutely, hourly or daily manner. If the requirements on the topicality of the data s lower, techniques such as caching and temporal data storage may be used [19][29][53][67].

Effects of the data changes: The influcence factor focus on the implications of data variations regarding the business success [29][30][32]. This factor can be interpreted as the sensitivity of economical effects if data changes. For instance, even small irregularities during the paint work in the automobile assembly lead to elaborate amendments in the subsequent production process.

Range of variation: To what extend does the performance measures change and what is the potential impact of data variations on the business [29][30][32]?

C. Characteristics of industry - economic factors

Internal realization conditions and implementation period: How quickly can the results and decisions from the IT system be implemented. The use of real-time data has no added value if, despite a rapid calculation, several days are required for the implementation of the results. The availability of real-time information makes it necessary to react quickly to changes. For the use of the entire potential, it is often necessary to comprehensively adapt the affected business processes in a company [23][32][36][38][39][63][53].

Potential (economic) added value: The acquisition of a new IT system entails high investment costs. This raises the question of whether the target market has enough potential to generate the necessary added value [61][62].

Target group acceptance: Are customers ready and willing to deliver all necessary data (e.g., GPS data)? On the other hand, it is also important to know if a customer is constantly accepting fluctuating prices in the supermarket [63].

Technical realization conditions: Is it actually possible to collect and transfer the desired data in due time under difficult conditions? For example, the transmission of sensor data from a Brazilian silvermine (or other remote locations) to the company's IT system [64].

Legal realization conditions: To what extent may data be evaluated at all or is the system applicable worldwide. In Germany, for example, there are very strict requirements regarding the data privacy and the processing of personal data. Legal issues in terms of data privacy may occur across borders or make once legal practices illegal in terms of the General Data Protection Regulation (GDPR) [68]. Another example is the decision against the permission of so called geoblocking and discrimination of EU online shop users [69].

VI. Application example of the framework

In this section the functionality of the framework will be shown based on selected application examples from different sectors. The examples were discussed during the expert interviews and in initial case studies. Based on the selected use cases, the advantages of our IMIS evaluation framework are clarified.

1) Case study "Analysis of Sales and Inventory Data": The analysis of sales and inventory data is one of the core tasks in the retail sector. One goal is the reduction of storage and delivery costs. At the same time out-of-stock situations should be avoided. Thereby, it is necessary to take current sales figures, fluctuations due to promotions as well as external influences into account. The main characteristics of this business process are described below:

Data characteristics: Due to the large number of sales transactions the underlying data volume is quite high. Sales documents in the retail sector are well-structured and can therefore be easily processed. A complex transformation is not necessary. The sales figures are subject to extensive and quite frequent variations. In practice, fluctuations can occur up to 500% [70]. This can lead to out-of-stock situations within a very short time. This in turn leads to a decreasing customer satisfaction and the loss of potential sales [71].

Analysis characteristics: Caused by the already mentioned variations of the sales figures it is necessary to recognize anomalies as fast as possible. Therefore, high requirements regarding the urgency of the analysis are formulated in this case. The analysis of the information is mostly based on recurring standard reports. The complexity of the evaluation as well as the complexity of the underlying data in this case study is typically low. To detect the anomalies it is sufficient to evaluate the current inventory information which is typically stored in transactional systems.

Economic characteristics: Due to the high range of fluctuations in sales figures, the fast detection of corresponding anomalies and the avoidance of out-of-stock situations leads to a better customer satisfaction and a economic benefit. At this point, the question arises which measures can be taken to minimize the fluctuations. In the work of Piller and Hagedorn [29], only non-price measures are proposed to reduce sales fluctuation. In this case, there are no legal or technical obstacles to the realization of the measures. As shown in Figure 4, not only the improved data availability and faster data processing play a role to generate business value, but also the downstream decision and implementation processes. In the example, "Analysis of Sales and Inventory Data," this issue represents the main weakness. Even with multiple daily supplies, deliveries may take several hours. This long implementation latency diminishes the advantages of an accelerated data and analysis processing.

Evaluation of the case study: Evaluating the presented characteristics, it becomes clear that in overall the data requirements are on a high level. This is caused in particular by the high data volume and high frequency of data changes. The analyses requirements can be summarized as low up to medium. Although the results need to be quickly processed they include only transactional tasks, the analysis are predictable and not particularly complex. From a technical perspective the use of an IMIS System is not essential. In combination with the economic limitations, traditional enterprise resource planing system are still able to monitor and control of stock levels.

2) Case study "Spatial Analytics in Soccer": As already mentioned in Section II-B the analysis of soccer was one of the first IMIS application scenarios and one of the most popular showcases. In this case, video and sensor data from soccer matches are evaluated and recommendations are given to the coaches. With the help of the evaluation model presented in this work, the potential for the use of spatial analysis in soccer should be assessed.

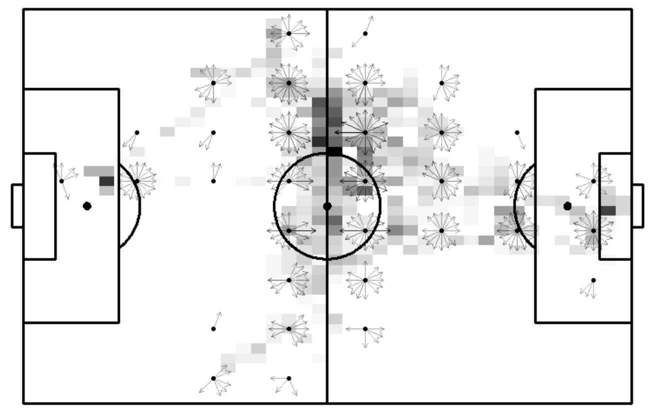


Figure 6. Example of a spatial analysis in soccer

Data characteristics: The processing of spatial data places very high demands on IT systems. In the case of spatial analysis in soccer this involves very large amounts of data with a complex structure [72]. The constant movement of the players on the field leads to a high frequency of data changes. A sample spatial analysis in soccer is visualized in a movement heatmap diagram in Figure 6. This example illustrates well the high demands on the data processing.

Analysis characteristics: The fixed time horizon of a soccer game demands, that the results of the analysis have to be provided within at least 90 minutes. Furthermore, the continuous changes in a game lead to a very fast decrease of the competitive advantage of the information. Most added value is therefore achieved by the provisioning in real-time. Another challenge is the processing of video recordings in conjunction with pattern recognition. This requires the use of advanced algorithms. The demanded degree of detail varies between individual and team analysis. All these points highlight the high requirements regarding the analysis capabilities.

Economic characteristics: From an economic point of view, competitive advantages through better and faster information promise a tremendous added value. The implementation of this case has shown that the required technical capabilities are available. It can be assumed that the players are able to realize the suggestions from the IT system. So far there are no limitations which interfere the use of spatial analysis in soccer. The limiting factors in this scenario are the legal reglementations. An extraction from the FIFA Statutes shows that such aids are not allowed during a soccer match: "information and data transmitted from the devices/systems is not permitted to be received or used in the technical area during the match" cited from [73]. The technical area depicts a designated space on the soccer field restricted to the technical staff, coaches and substitute players. This restriction severely limits the potential added value. As a result, the analysis can only be used for the post-processing and the preparation of games.

Evaluation of the case study: The evaluation of the attributes from a technical point of view implies the suitability of an IMIS. The data as well as the analysis characteristics require a sophisticated IT system. Due to the permanent recording of the spatial information, very large amounts of data accumulate. Conventional information systems can not complete these tasks in sufficient time. The benefit of the presented framework becomes especially clear when the economic factors are considered. Basically, a faster and more flexible processing would lead to a high economic benefit. The major benefit in this use case is limited through legal reglementations. These reglementations result in an forced implementation latency of several hours. Within this time horizon also conventional IT systems are able to perform the calculations. An IMIS system is therefore not mandatory.

3) Case study "Management of Renewable Energy": The energy revolution and the closely related use of renewable energy resources is one of the greatest challenges facing states, companies and individuals today. The scenario to be analyzed comprise a holistic "energy cycle" - from the generation to the transmission up to the consumer. The concept of a connected intelligent energy network is often called 'smart grid'. The efficient execution of this holistic energy process requires the consideration of numerous factors such as the weather, the energy grid utilization or the current consumption. The process characteristics are described below analogously to the previous examples:

Data characteristics: In the energy sector very large amounts of data accrue every day. These data is generated e.g., by smart meter devices, weather stations or the continous application of sensors. This variety of data sources leads to an increasing of the data complexity. The data processing increasingly requires the consideration of complex data structures [74]. Another challenging characteristic is the high data dynamic. The amount of the produced energy, the energy demand, etc. are frequently changing. In addition to the high frequency of changes the impact of data changes is substantial. All in all the demands placed on data processing can be assessed as high in the examined criteria.

Analysis characteristics: The efficient control of measures requires, that the analysis results have to be available in real-time. In this use case, current data from a variety of source systems must be taken into account. Analogous to the previous case this results in an interaction of transactional and analytical tasks. An important part of the management of renewable energy is the prediction of the produced energy. Thereby complex procedures are used, considering for example weather conditions [75]. Similar to the data requirements there are high analysis requirements.

Economic characteristics: As already described in the introduction of this use case, this scenario offers comprehensive possibilities for improvements. The potential added value comprises beside monetary savings also positive ecological aspects. Through the use of smart meter devices, controllable energy networks and improved weather forecasts the technical requirements for smart grids are already in place. The only weak point in the economic evaluation of this scenario is the inconsistent user acceptance. Studies in this field have revealed a differing acceptance to adopt to this technology [76].

Evaluation of the case study: The technical requirements in this case are very high in all areas. The large data volume, the data volatility, recently changing data, the need for quick responses and the combination of analytical and transactional tasks are strong indicators for the suitability of an IMIS. The economic analysis of this example also shows a high added value through faster data processing. Furthermore, the realization conditions indicates no basic obstacles. The model therefore indicates the suitability of the In-memory technology in all examined categories.

VII. Conclusion

The aim of this work was to create a framework for analyzing and evaluating application scenarios in the context of IMIS. As current research as well as statements from industry experts show, such a framework was missing. To cover all relevant factors for the application of an IMIS, not only theoretical work was included in this work. Through the inclusion of corporate experts, also practical aspects have been considered. Based on the first case studies in this area and scientific work, a large part of the influencing factors could be identified. Results show that the influence factors found through literature review and expert study could be divided into three main categories: analysis factors, data-driven factors and economic factors. Based on the expert survey, it was also possible to confirm the factors from the literature and to uncover other previously unconsidered factors. In order to take account of all aspects relevant to the companies, the model was expanded by features with regard to the profitability and the feasibility of possible fields of application. These include, for instance, the implementation conditions, legal obstacles or the acceptance of target groups. The capabilities of the framework were emphasized through the presented cross-industry use cases. It became clear that the evaluation of possible IMIS application scenarios requires a holistic view of all aspects. The presented model can be used as an additional assessment instrument for corporate decision makers.

In a next step, it will be necessary to evaluate the suitability of the framework based on quantitative investigations in different industry sectors. The use cases presented in this work indicate a relevance variation in terms of the assessment of the influence factors. It is supposed that the relevance of the influcence factors varies between branches and companies. Another question not considered in this work is the implicit and explicit nature of the influence factors in terms of value creation. Some factors like urgency have a very direct explicit impact on the value creation whereas other factors such as data complexity or data volume have a more indirect and implicit impact. Consequently, this fact should be taken into account in future work.

References

[1] S. Ulbricht, M. Opuszko, J. Ruhland, and M. Thrum, "Towards an analysis and evaluation framework for in-memory-based use cases," in The Twelfth International Multi-Conference on Computing in the Global Information Technology, 2017, pp. 22–27.

[2] "Amazon Introduces Prime Now," 2014, URL: http://www.businesswire.com/multimedia/home/20141218005363/en/ [accessed: 2017-03-14].

[3] "The Wall Street Journal - High-Speed Stock Traders Turn to Laser Beams," 2014, URL: http://www.wsj.com/articles/SB10001424052702303947904579340711\\424615716 [accessed: 2017-03-14].

[4] H. Rosa, Beschleunigung: die Veränderung der Zeitstrukturen in der Moderne, ser. Suhrkamp Taschenbuch Wissenschaft. Suhrkamp, 2005.

[5] "SAP fehlen echte HANA-Business-Cases," 2015, URL: http://www.cio.de/a/sap-fehlen-echte-hana-business-cases,2940526 [accessed: 2017-03-14].

[6] "Lack of SAP HANA use cases stifling demand among ASUG members," 2014, URL: http://diginomica.com/2014/08/08/lack-sap-hana-use-cases-stifling-demand-among-asug-members/ [accessed: 2017-03-14].

[7] "ASUG Member Survey Reveals Successes, Challenges of SAP HANA Adoption," 2014, URL: http://www.asugnews.com/article/asug-member-survey-reveals-successes-challenges-of-sap-hana-adoption [accessed: 2017-03-14].

[8] "Hunting happy HANA customers," 2014, URL: http://diginomica.com/2014/10/27/hunting-happy-hana-customers/ [accessed: 2017-03-14].

[9] "SAP Business Suite powered by SAP HANA," 2014, URL: https://www.pac-online.com/download/9757/125462 [accessed: 2017-03-09].

[10] H. Garcia-Molina and K. Salem, "Main memory database systems: An overview," IEEE Transactions on knowledge and data engineering, vol. 4, no. 6, 1992, pp. 509–516.

[11] D. J. DeWitt, R. H. Katz, F. Olken, L. D. Shapiro, M. R. Stonebraker, and D. A. Wood, Implementation techniques for main memory database systems. ACM, 1984, vol. 14, no. 2.

[12] F. Färber, S. K. Cha, J. Primsch, C. Bornhövd, S. Sigg, and W. Lehner, "Sap hana database: data management for modern business applications," ACM Sigmod Record, vol. 40, no. 4, 2012, pp. 45–51.

[13] H. Plattner and A. Zeier, In-memory data management: technology and applications. Springer Science & Business Media, 2012.

[14] M. K. Gupta, V. Verma, and M. S. Verma, "In-memory database systems-a paradigm shift," 2014, pp. 333–336.

[15] D. J. Abadi, P. A. Boncz, and S. Harizopoulos, "Column-oriented database systems," Proceedings of the VLDB Endowment, vol. 2, no. 2, 2009, pp. 1664–1665.

[16] A. Kemper and T. Neumann, "Hyper: A hybrid oltp&olap main memory database system based on virtual memory snapshots," in Data Engineering (ICDE), 2011 IEEE 27th International Conference on. IEEE, 2011, pp. 195–206.

[17] H. Plattner, "A common database approach for oltp and olap using an in-memory column database," 2009, pp. 1–2.

[18] N. May, A. Böhm, and W. Lehner, "Sap hana–the evolution of an in-memory dbms from pure olap processing towards mixed workloads," Datenbanksysteme für Business, Technologie und Web (BTW 2017), 2017.

[19] P. Loos, J. Lechtenbörger, G. Vossen, A. Zeier, J. Krüger, J. Müller, W. Lehner, D. Kossmann, B. Fabian, O. Günther et al., "In-memory databases in business information systems," Business & Information Systems Engineering, vol. 3, no. 6, 2011, pp. 389–395.

[20] M. Pezzini, D. Feinberg, N. Rayner, and R. Edjlali, "Hybrid transaction/analytical processing will foster opportunities for dramatic business innovation," Gartner (2014, January 28) Available at https://www. gartner. com/doc/2657815/hybrid-transactionanalytical-processing-foster-opportunities, 2014.

[21] J. Krueger, M. Grund, C. Tinnefeld, B. Eckart, A. Zeier, and H. Plattner, "Hauptspeicherdatenbanken für unternehmensanwendungen," Datenbank-Spektrum, vol. 10, no. 3, 2010, pp. 143–158.

[22] P. Loos, S. Strohmeier, G. Piller, and R. Schütte, "Comments on in-memory databases in business information systems," Business & Information Systems Engineering, vol. 4, no. 4, 2012, pp. 213–223.

[23] S. Strohmeier, "Hauptspeicherdatenbanken in der betrieblichen informationsversorgung–technische innovation und fachliche stagnation," Wirtschaftsinformatik, vol. 54, no. 4, 2012, pp. 209–210.

[24] J. Krueger, M. Grund, C. Tinnefeld, H. Plattner, A. Zeier, and F. Faerber, "Optimizing write performance for read optimized databases," in Database Systems for Advanced Applications. Springer, 2010, pp. 291–305.

[25] J. Wust, J.-H. Boese, F. Renkes, S. Blessing, J. Krueger, and H. Plattner, "Efficient logging for enterprise workloads on column-oriented in-memory databases," in Proceedings of the 21st ACM international conference on Information and knowledge management. ACM, 2012, pp. 2085–2089.

[26] A. K. Goel, J. Pound, N. Auch, P. Bumbulis, S. MacLean, F. Färber, F. Gropengiesser, C. Mathis, T. Bodner, and W. Lehner, "Towards scalable real-time analytics: an architecture for scale-out of olxp workloads," Proceedings of the VLDB Endowment, vol. 8, no. 12, 2015, pp. 1716–1727.

[27] D. Abadi, S. Madden, and M. Ferreira, "Integrating compression and execution in column-oriented database systems," in Proceedings of the 2006 ACM SIGMOD international conference on Management of data. ACM, 2006, pp. 671–682.

[28] T. Winsemann and V. Köppen, "Kriterien für datenpersistenz bei enterprise data warehouse systemen auf in-memory datenbanken." in Grundlagen von Datenbanken, 2011, pp. 97–102.

[29] G. Piller and J. Hagedorn, "In-memory data management im einzelhandel: Einsatzbereiche und nutzenpotentiale," Multikonferenz Wirtschaftsinformatik 2012 : Tagungsband der MKWI 2012 / Hrsg.: Dirk Christian Mattfeld; Susanne Robra-Bissantz.

[30] ——, "Business benefits and application capabilities enabled by in-memory data management," in Innovative Unternehmensanwendungen mit In-Memory Data Management, IMDM 2011, 2. Dec 2011, Mainz, ser. LNI, W. Lehner and G. Piller, Eds., vol. 193. GI, 2011, pp. 45–56.

[31] ——, "Einsatzpotenziale für in-memory data management in betrieblichen anwendungssystemen," Wirtschaftsinformatik & Management, vol. 3, no. 5, 2011, pp. 18–25.

[32] C. Cundius and R. Alt, "Real-time or near real-time?-towards a real-time assessment model," Thirty Fourth International Conference on Information Systems, 2013, pp. 1–18.

[33] ——, "A process-oriented model to business value–the case of real-time it infrastructures," in Proceedings of the 50th Hawaii International Conference on System Sciences, 2017.

[34] B. Otto and R. Bärenfänger, "Potentiale und risiken von in-memory-technologie: Eine business engineering-perspektive," in Wirtschaftsinformatik in Wissenschaft und Praxis. Springer, 2014, pp. 153–163.

[35] M. Nadj and C. Schieder, "Towards a taxonomy of real-time business intelligence systems," 2017.

[36] J. vom Brocke, S. Debortoli, and O. Müller, "In-memory database business value," 360 - The Business Transformation Journal, vol. 3, no. 7, 2013, pp. 16–26.

[37] J. vom Brocke, "In-memory value creation, or now that we found love, what are we gonna do with it?" BPTrends, vol. 10, 2013, pp. 1–8.

[38] J. vom Brocke, S. Debortoli, O. Müller, and N. Reuter, "How in-memory technology can create business value: insights from the hilti case," Communications of the Association for Information Systems, vol. 34, no. 1, 2014, pp. 151–167.

[39] R. Bärenfänger, B. Otto, and H. Österle, "Business value of in-memory technology–multiple-case study insights," Industrial Management & Data Systems, vol. 114, no. 9, 2014, pp. 1396–1414.

[40] M. C. Meier, A. Scheffler, and K. Finanz, "Ökonomisch sinnhafte bewertung von in-memory-basierten betrieblichen informationssystemen." in IMDM. Citeseer, 2011, pp. 115–124.

[41] R. Winter, S. Bischoff, and F. Wortmann, "Revolution or evolution? reflections on in-memory appliances from an enterprise information logistics perspective." in IMDM, 2011, pp. 23–34.

[42] R. Schütte, "Analyse des einsatzpotenzials von in-memory-technologien in handelsinformationssystemen." in IMDM, 2011, pp. 1–12.

[43] R. Meyer, V. Banova, A. Danciu, D. Prutscher, and H. Krcmar, "Assessing the suitability of in-memory databases in an enterprise context," in Enterprise Systems (ES), 2015 International Conference on. IEEE, 2015, pp. 78–89.

[44] "McLaren Formula 1 - Partners - SAP," 2015, URL: http://www.mclaren.com/formula1/partners/SAP [accessed: 2017-03-14].

[45] "Big Data & Spatial Analytics Help Germany Score the World Cup," 2014, URL: http://www.saphana.com/community/blogs/blog/2014/07/15/how-big-data-helped-germany-score-the-world-cup [accessed: 2017-03-14].

[46] G. Koleva, "Fields of usage for in-memory databases in enterprises," in 11th Workshop on Information Systems and Services Sciences, 2011, pp. 19–33.

[47] A. R. Hevner, S. T. March, J. Park, and S. Ram, "Design science in information systems research," MIS quarterly, vol. 28, no. 1, 2004, pp. 75–105.

[48] A. R. Hevner, "A three cycle view of design science research," Scandinavian journal of information systems, vol. 19, no. 2, 2007, p. 4.

[49] R. Klein and A. Scholl, Planung und Entscheidung. Vahlen, München, 2004.

[50] R. Hackathorn, "Minimizing action distance," DM REVIEW, vol. 12, 2002, pp. 22–23.

[51] J. Webster and R. T. Watson, "Analyzing the past to prepare for the future: Writing a literature review," MIS quarterly, 2002, pp. xiii–xxiii.

[52] H. O. Mayer, Interview und schriftliche Befragung: Grundlagen und Methoden empirischer Sozialforschung. Walter de Gruyter, 2013.

[53] H. Zhang, G. Chen, B. C. Ooi, K.-L. Tan, and M. Zhang, "In-memory big data management and processing: A survey," IEEE Transactions on Knowledge and Data Engineering, vol. 27, no. 7, 2015, pp. 1920–1948.

[54] M. Nadj and C. Schieder, "Quo vadis real-time business intelligence? a descriptive literature review and future directions," 24th European Conference on Information Systems, 2016, pp. 1–20.

[55] W. H. Inmon, Building the data warehouse. John wiley & sons, 2005.

[56] L. Berti-Equille and J. Borge-Holthoefer, "Veracity of data: From truth discovery computation algorithms to models of misinformation dynamics," Synthesis Lectures on Data Management, vol. 7, no. 3, 2015, pp. 1–155.

[57] H. Baars and H.-G. Kemper, "Management support with structured and unstructured data an integrated business intelligence framework," Information Systems Management, vol. 25, no. 2, 2008, pp. 132–148.

[58] B. Inmon and K. Krishnan, Building the Unstructured Data Warehouse: Architecture, Analysis, and Design. Technics Publications, 2011.

[59] T. Knabke and S. Olbrich, "Grundlagen und einsatzpotentiale von in-memory-datenbanken," in Analytische Informationssysteme. Springer, 2016, pp. 187–203.

[60] S. Sarawagi, "Queries over unstructured data: Probabilistic methods to the rescue," in International Workshop on Business Intelligence for the Real-Time Enterprise. Springer, 2009, pp. 1–13.

[61] L. Olsson and C. Janiesch, "Real-time business intelligence und action distance: Ein konzeptionelles framework zur auswahl von bi-software." in Wirtschaftsinformatik, 2015, pp. 691–705.

[62] R. Hackathorn, "Real-time to real-value," Information Management, vol. 14, no. 1, 2004, p. 24.

[63] M. Bauer, Kundenzufriedenheit in industriellen Geschäftsbeziehungen: kritische Ereignisse, nichtlineare Zufriedenheitsbildung und Zufriedenheitsdynamik. Springer-Verlag, 2013.

[64] S. King, Big Data: Potential und Barrieren der Nutzung im Unternehmenskontext. Springer-Verlag, 2014.

[65] S. Ciriani, "The economic impact of the european reform of data protection," Digiworld Economic Journal, vol. 97, 2015, pp. 41–58.

[66] W. Sinzig and K. R. Sharma, "In-memory-technologie: Verbesserungen bei planung, simulation und entscheidungsunterstützung," Wirtschaftsinformatik & Management, vol. 3, no. 2, 2011, pp. 18–23.

[67] J. Lee, Y. S. Kwon, F. Frber, M. Muehle, C. Lee, C. Bensberg, J.-Y. Lee, A. H. Lee, and W. Lehner, "Sap hana distributed in-memory database system: Transaction, session, and metadata management." in ICDE, C. S. Jensen, C. M. Jermaine, and X. Zhou, Eds. IEEE Computer Society, 2013, pp. 1165–1173. [Online]. Available: http://dblp.uni-trier.de/db/conf/icde/icde2013.html#LeeKFMLBLLL13

[68] M. Körner, "Die reform des eu-datenschutzes: Der entwurf einer eu-datenschutz-grundverordnung (ds-gvo)–teil ii," 2018.

[69] J. Borer, "Spruchpraxis zum eu-wettbewerbsrecht (2016/2017)," Swiss. Rev. Int'l & Eur. L., vol. 27, 2017, p. 251.

[70] C. Narasimhan, S. A. Neslin, and S. K. Sen, "Promotional elasticities and category characteristics," The Journal of Marketing, 1996, pp. 17–30.

[71] P. Verhoef and L. Sloot, "Out-of-stock: reactions, antecedents, management solutions, and a future perspective," Retailing in the 21st Century, 2006, pp. 239–253.

[72] A. Bialkowski, P. Lucey, P. Carr, Y. Yue, S. Sridharan, and I. Matthews, "Large-scale analysis of soccer matches using spatiotemporal tracking data," in Data Mining (ICDM), 2014 IEEE International Conference on. IEEE, 2014, pp. 725–730.

[73] "Laws of the Game 2017-18," 2017, URL: http://resources.fifa.com/mm/document/footballdevelopment/refereeing/02/90/11/67/lawsofthegame2017-2018-en_neutral.pdf [accessed: 2018-02-09].

[74] K. Zhou, C. Fu, and S. Yang, "Big data driven smart energy management: From big data to big insights," Renewable and Sustainable Energy Reviews, vol. 56, 2016, pp. 215–225.

[75] P. D. Diamantoulakis, V. M. Kapinas, and G. K. Karagiannidis, "Big data analytics for dynamic energy management in smart grids," Big Data Research, vol. 2, no. 3, 2015, pp. 94–101.

[76] M. Broman Toft and J. Thøgersen, "Exploring private consumers willingness to adopt smart grid technology," International journal of consumer studies, vol. 39, no. 6, 2015, pp. 648–660.

On the Development of a Cyber-Physical Industrial Marketplace

Susana Aguiar, Rui Pinto, João Reis, Gil Gonçalves

Department of Informatics Engineering
Faculty of Engineering, University of Porto
Porto, Portugal
Email: {saguiar, rpinto, jpcreis, gil}@fe.up.pt

***Abstract*—In recent years, we have seen breathtaking technological progress within Cyber-Physical Systems (CPS), the Internet of Things (IoT), Cloud computing, and intelligence and data analytics. These advances are indeed moving from research to the industrial shop floor and paving the way for the fourth industrial revolution, the Industry 4.0. Industry 4.0 is putting technology to work, disrupting conventional industrial processes at every level and driving the industry into becoming digital. All the technological progress has allow knowledge and information to be virtually available everywhere to anyone, leading to the emergence of new possibilities and business models. In a world where competition is global, the industry must be able to respond to unpredictable and rapid changes. The use of the Smart Component concept, integrated into the Cyber-Physical System architecture, allows for available data to be processed and used to improve productivity and production planning as well as allow for a better equipment use and better predictive maintenance practices. All this available data must be stored and accessible, which is where the Cyber-Physical Industrial Marketplace comes into play, providing an interface for automatic update of the industrial equipment technologies and functionalities on the fly. In this paper, an architecture for a Cyber-Physical Industrial Marketplace is presented and explored. Having the ability to collect data from the shop-floor equipment and update the Smart Components in a safe, reliable, and secure way, is the main goal of the proposed architecture. Creating an industrial marketplace that goes beyond the concept of online stores, is the next natural and logical step that will contribute to the servitisation of industrial business models.**

Keywords–Smart Factories; Intelligent Production Systems; Industry 4.0; Cyber-Physical Systems; Marketplace.

I. INTRODUCTION

Smart Factories, Cyber-Physical Systems (CPS), Internet of Things (IoT), and Cloud computing are popular terms that are nowadays in the spotlight of current technology advances. These technologies offer many advantages and are at the core of the Industry 4.0 concept, which was at the basis of the Marketplace for Cyber-Physical Production Systems architecture presented in [1].

Industry 4.0: The term Industry 4.0, also recognized as the fourth industrial revolution, originated in Germany, from a project promoted by the German government with the goal of digitizing the manufacturing industry [2]–[5]. The first industrial revolution began in England in the late 18th century with the introduction of mechanical production equipment driven by water and steam power. The second industrial revolution started in the beginning of the 20th century and was based on mass production enabled by the division of labour and the use of electrical energy. In the beginning of the 1970's, the third industrial revolution was initiated with the use of electronics and IT to further automate production. This lead to today's industrial revolution, the fourth industrial revolution - Industry 4.0, with the use of CPS. With the Industry 4.0 new concepts and business models have emerged (Figure 1) [6]–[11].

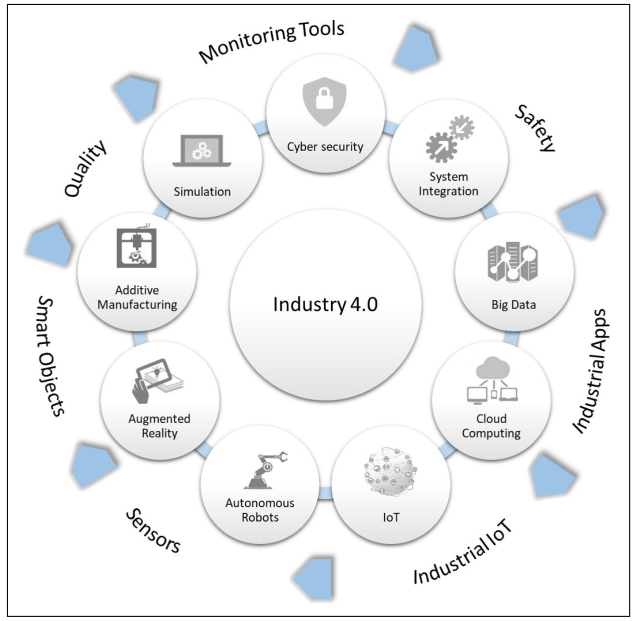


Figure 1. Industry 4.0

There are several key factors that must be present in a factory or in a system to be Industry 4.0 compliant: interoperability, information transparency, technical assistance, and decentralized decision making. Interoperability gives devices, machines, sensors, and humans the ability to connect and communicate with each other. Information transparency is achieved through the creation of a virtual copy of the physical world using the sensor data in order to contextualize information. Technical assistance has two main objectives: support humans in decision making and problem solving, and assist humans with tasks that are too difficult or too dangerous for humans. Decentralized decision making is reached through passing simple decisions to the CPS level, which becomes as autonomous as possible.

The Industry 4.0 has also created multiple challenges [12], [13], such as security, reliability and stability, integrity of the production system, loss of jobs, and fear of new technologies. Data security, in any business, is always a concern that is hugely increased when new systems and new ways of accessing the systems are introduced, which consequently also creates new risks of losing proprietary information technology

(IT) production knowledge. Although technology has come a long way, there are still some limitations to overcome, and for a CPS system to be successful, it must be reliable and stable, which can sometimes be difficult to achieve and to maintain. Having a production system with less human supervision can also be a barrier if integrity is not guaranteed, and less human jobs always lead to concerns in the work environment.

The manufacturing industry has traditionally been a top-down and centralized planning process. This way of working is fast becoming insufficient to respond to the constant changes in the new high-mix low-volume production environments. This is another reason for the strength that the Industry 4.0 concept has gain, and one of the ways to cope with these new production environments is through reconfigurability, which is facilitated by the technologies behind the Industry 4.0.

The concept of reconfigurability has been around for many years. Generally, reconfigurability can be though as the ability to repeatedly change and rearrange the components of a system in a cost-effective way. Koren *et al.* in [14] defined reconfigurable manufacturing systems as being "[..] designed at the outset for rapid change in structure, as well as in hardware and software components, in order to quickly adjust production capacity and functionality [..] in response to sudden changes in market or in regulatory requirements". Merhabi *et al.* [15] complement this definition with the notion that "re-configuration allows adding, removing or modifying specific process capabilities, controls, software, or machine structure to adjust production capacity in response to changing market demands or technologies [..] provides customised flexibility [..] so that it can be improved, upgraded and reconfigured, rather than replaced".

The equipment lifetime traditionally has several stages: it starts with its incorporation onto the production line, followed by its effective operation and maintenance phase, to its end use and consequent removal. It is possible through these steps to identify several critical points of intersection, which can potentially cause equipment downtime and possibly downtime of the production line, which ultimately is reflected in additional costs.

To aid the reconfigurability in the manufacturing industry, smart objects or smart components are being developed and used. Smart Components are defined as components that incorporate functions of self-description, communication, sensing, and control in order to cooperate with other smart components, analyse a situation, make decisions based on the available data, and modify their behaviour through feedback [16], [17].

Cloud Services: All the sensors and CPS that are being used and can be used in the industry, produce massive amounts of data that can be used to further improve the manufacturing process. The challenge is where and how to store all this data and how to use it. One way of dealing with this challenge is to use cloud based services [18]–[22]. The term "cloud" started to be used in the 1990's in the telecommunications field, when providers began using virtual private network (VPN) services for data communication [23], [24]. The National Institute of Standards and Technology (NIST) defines cloud computing as: "a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage applications and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction." [25].

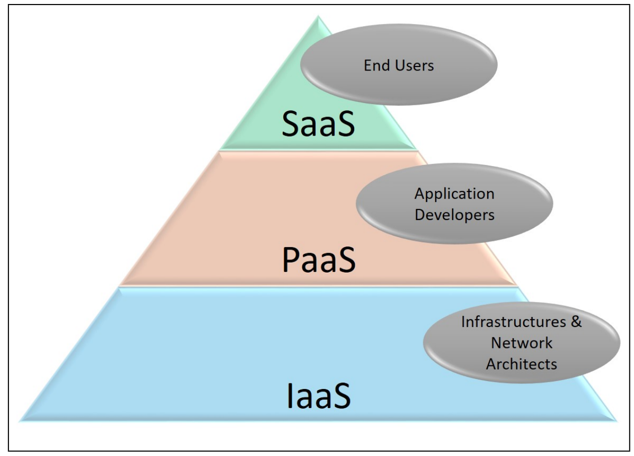


Figure 2. Cloud Services Model

The cloud computing concept provides three types of services that can be arranged as layers in a stack 2: the Software as a Service (SaaS), the Platform as a Service (PaaS), and the Infrastructure as a Service (IaaS). SaaS provide the user with the capability of running applications that are hosted in the cloud instead of locally, which can be accessed from various client devices. PaaS allow the user to deploy, to the cloud, user-created or acquired applications created using programming languages, libraries, services, and tools supported by the provider. IaaS provide the capability to provision processing, storage, networks, and other fundamental computing resources where the user is able to deploy and run arbitrary software, which can include operating systems and applications [24], [26]–[28].

As mentioned before, new business models are emerging in order to accommodate all these new concepts, technologies, and changes. One of these business models is what can be called of Industrial Marketplace. An Industrial Marketplace is a Multi-sided platform (MSP), also known as virtual Marketplace, [29], [30]. There are several definitions of what a MSP is, which are not always in accordance with each other [29], but there are two key features that are always present: (1) platforms, which provide the ability of direct interactions between two or more distinct sides, and (2) each side is affiliated with the platform. Furthermore, in order for a platform to be successful it must provide the ability of co-creation of value, interdependency and complementarity of components, surplus value for the whole system, and evolutionary growth [31].

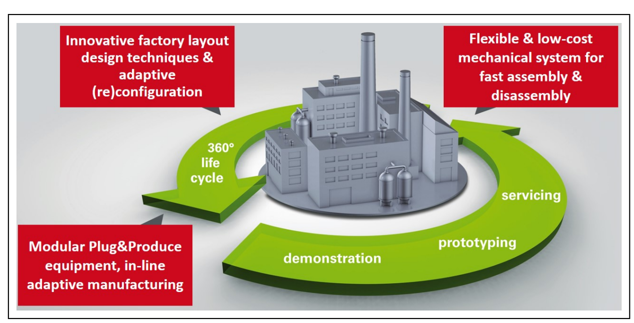


Figure 3. ReBorn Vision

ReBorn Project: As presented in [1], the architecture described is the result of combining the outcomes of two European projects: the Reborn project and the Selsus project. The ReBorn - Innovative Reuse of modular knowledge Based devices and technologies for Old, Renewed and New factories project [32], was a project funded under THEME FoF.NMP.2013-2 - Innovative re-use of modular equipment based on integrated factory design until August of 2016. The vision of ReBorn was to demonstrate strategies and technologies that support a new paradigm for the re-use of production equipment in factories (Figure 3). This re-use will give new life to decommissioned production systems and equipment, helping them to be "reborn" in new production lines. Such new strategies will contribute to sustainable, resource-friendly and green manufacturing and, at the same time, deliver economic and competitive advantages for the manufacturing sector. The developments made in ReBorn helped in the validation of technologies that extends production equipment life cycle, contributing to economic and environmental sustainability of production systems [9], [33].

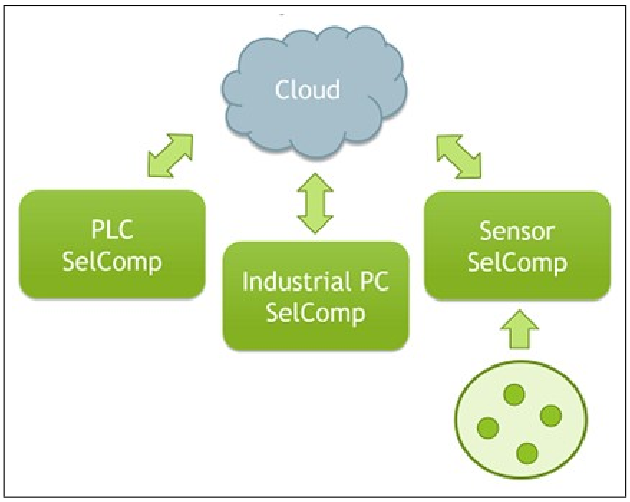


Figure 4. SelSus Vision

Selsus Project: The SelSus - Health Monitoring and Life-Long Capability Management for SELf-SUStaining Manufacturing Systems project [34] was a project also funded by the European Commission under the Seventh Framework Program for Research and Technological Development until August of 2017. The vision of SelSus was to create a new paradigm for highly effective, self-healing production resources and systems to maximize their performance over longer life times through highly targeted and timely repair, renovation and upgrading through the use of the Smart Component concept as a SelComp (Figure 4). These next generation machines, fixtures and tools with embed extended sensory capabilities and smart materials combined with advanced Information and Communications Technology (ICT) for self-diagnosis enabling them to become self-aware and supporting self-healing production systems. Distributed diagnostic and predictive repair and renovation models will be embedded into smart devices to early prognosis failure modes and component degradations. Self-aware devices will built on synergetic relationships with their human operators and maintenance personnel through continuous pro-active communication to achieve real self-healing systems. This will drastically improve the resilience and long term sustainability of highly complex manufacturing facilities to foreseen and unforeseen disturbances and deteriorations thereby minimizing energy and resource consumption and waste [17], [35]–[37].

In this paper, we propose an architecture for a Cyber-Physical Industrial Marketplace that results from combining the experience collected over the development of the ReBorn Marketplace, the SelSus Dashboard, and the combination of the functionalities of the Smart Components implemented on both projects. The objective is to have an efficient and easy to use Industrial Marketplace that provide a route for the flow of information from the equipment at the shop-floor, up to the Marketplace and back. Further, this proposed Cyber-Physical Industrial Marketplace will allow us to take advantage of all the current technological advances, and contributes for a safe and reliable way of using all the available information. Section II provides an overview of related work. Section III presents an overview of an Industrial Marketplace concept and Section IV describes the proposed Cyber-Physical Industrial Marketplace concept. Finally, Section V concludes the paper by exposing some final remarks about the concept and work developed.

II. Related Work

A digital marketplace can be viewed as a collaborative network that is formed in order to take advantage of new opportunities and explore new business models, as mentioned before. Nowadays, processing sensors and actuators have become common, more affordable and available, which is driving the wide adoption of Wireless Sensor Networks (WSNs) solutions. The WSNs can be found practically in all areas of everyday life, in applications such as environmental monitoring, military, or industrial fields, specially since data gathering is one of the pillars for the implementation of the IoT concept. Since WSNs typically contain small sensor devices, they have limitations on the memory, computation, energy, and scalability [38]. In order to fully use the potentialities of the WSNs, an infrastructure that is powerful, scalable, and secure must be implemented.

Llanes *et al.* [39] presented a survey of the main approaches that have been developed to deal with all the raw data collected by sensors. Sensors continuously collect data regarding a given event and send it to a gateway, which usually needs a specific protocol to process the received raw data. The main problem is that the various sensor manufacturers provide different communication protocols that use different message formats. This means that there is not a universal technology that can receive raw sensor data and support every message type for all the manufacturers. In the survey several solutions are described as well as the strengths and limitations of each one. An attempt to tackle this problem was made by Gonçalves *et al.* [40], where a Universal Gateway is proposed and presented. Shen *et al.* [41] provided an overview of recent developments of agent technology applied to manufacturing enterprises, which include enterprise collaboration regarding supply chain management and virtual enterprises, manufacturing process planning and scheduling, shop floor control, and also holonic manufacturing as an implementation methodology.

There have been other studies on how to manage the physical sensors. The Sensor Modelling Language (SensorML) [42] intends to provide standard models in a XML encoding for physical sensors' description and measurement processes. It is being used by the international non profit organization

Open Geospatial Consortium (OGC) [43] which is committed on making quality open standards for the global geospatial community.

As mentioned before, sensors from different manufacturers use different communication protocols, which makes sharing sensors and its information between applications difficult. Shneidman *et al.* [44] presented an infrastructure called Hourglass, which addresses the need for a software infrastructure that enables the rapid development and deployment of applications that use data from several, heterogeneous sensor networks. Silva *et al.* in [35] presented a flexible sensor integration solution. This flexible integration allows for a rapid graphical development of interpreters of raw data packets in the Cloud as well as its deployment for embedded execution at the WSN gateway level, for automatic data acquisition. Yuriyama & Kushida [45] proposed a new infrastructure called Sensor-Cloud infrastructure which can manage physical sensors on an IT infrastructure. The proposed Sensor-Cloud Infrastructure virtualizes a physical sensor as a virtual entity in the Cloud. Reis in [37] presented the implementation and use of the Sensor Cloud concept developed in the scope of the SelSus project. The goal was the easy integration, processing and visualization of sensor information within the industrial field. The main functionalities are the easy integration of sensors into a WSN regardless of their manufacturer, methods to pre-process data to be used for further decision making, and integration of new methods for statistical analysis.

This new paradigm of connecting and virtualizing sensors in a cloud infrastructure for data processing, is an active research field that is being explored by several researchers. Yan *et al.* [46], [47] proposed a cloud-based production system, across distributed data centers, which integrates several web and cloud computing technologies. A full connection model of product design and manufacturing in an IoT-enabled Cloud manufacturing environment, which uses the social networks to enable the connection of multiple parties, is also proposed. Zhang *et al.* [48] described a cloud manufacturing, defined for solving the bottlenecks in data and manufacturing applications. Alam & Saddik [49] presented and described a digital twin architecture reference model for cloud-based CPS, named C2PS. Neto *et al.* [50] presented the first steps in the development of a framework that takes advantage of several technologies like UPnP, OSGi, and iPOJO, which addresses some of the challenges needed to enable a Sensor Cloud in the shop floor. Chiang & Lee [51] proposed a smart manufacturing platform that can be used by small enterprises and start-ups. The heart of this platform is the communication pipeline, which allows component providers, original equipment manufacturer (OEM) factories, makers, and maker spaces that allow product development, to connect.

With so many solutions proposed and with so many different technologies that can be used, researchers have aggregate some of the available information through surveys. Alamri *et al.* [52] provided a survey of some of the most relevant work related to Sensor-Cloud infrastructure, its definition, architecture, and applications. Moçano *et al.* [53] analysed how the IoT can be used in the manufacturing industry, by proposing a metamodel for integrating the Internet of Things, Social Networks, Cloud, and Industry 4.0. Perera *et al.* [54] presented a survey over one hundred IoT smart solutions in the marketplace, identifying the technologies used, functionalities, and applications. The idea was to provide a guideline and a conceptual framework for future research in the IoT, motivating further developments by suggesting a number of potentially significant research directions. Albrecht *et al.* [55] examined the standards required for successful e-commerce architectures, evaluated the strengths and limitations of these type of systems and concluded, through the examination of the major platforms that have been developed, that there is a lack of a common or shared standard for marketplace architectures.

With the increase of the number of devices connected to the Internet, having centralized Cloud services will become unsustainable. This is leading to new paradigms such as Fog or Edge computing [56], [57].

Security and data privacy is currently one of the biggest restraints in the wide use of the technologies described previously. Many authors have addressed these issues of security and privacy over the IoT [58]–[64] and have proposed new approaches for securing and enabling reliability on sensor data [38], [65]. Nevertheless, there is still a great deal of work to be performed in areas such as cryptographic mechanisms, data, identity, and privacy management, as well as defining trusted architectures. Taylor & Sharif in [66] presented the main security threats faced by the industry as well as the main approaches for counteracting those threats. Zahra *et al.* in [67] also presented a concise survey on sensor network constraints, security requirements, attacks, and defensive measures. A RC5's algorithm was proposed, which provides good security against the four main types of attacks, even thought it is a simple encryption algorithm that still needs further testing.

The most common type of industrial marketplaces that can currently be found online are auction sites ([68], [69]) or digital stores, which are sites where companies present their products to be sold, like a normal store [70]–[72]. But there are also some companies trying to change their marketplace sites and make them more intelligent and take advantage of all the new technology available. One example is the Intelligent Plant company [73] that developed an app (App Store connect), which allows users that use their services and software to make their data available to the applications that are in their Industrial App Store. Another example is Advantech, which launched the WISE-PaaS Marketplace [74], an online IoT software store that features Advantech's exclusive software services, including IoT cloud services, IoT security services, and pre-packaged solution packages.

III. Industrial Marketplace

Although there are several platforms proposed for industrial marketplaces, there is still no truly intelligent, online, working industrial marketplace. There are several reasons for the lack of this type of Cyber-Physical Industrial Marketplace, which include security issues, a common accepted platform, and a common accepted communication mechanism. Another reason is trust. Most of the businesses in the industrial field are based on relationships built over time and where there is a high trust between the stakeholders [75].

The global marketplace has been evolving at a rapid pace over the recent years. Reduced lot sizes, customization and individualization in the product is the future of manufacturing. The challenge for manufacturers in this new trend of manufacturing is that neither the traditional theories of production planning and scheduling work in an Industry 4.0 environment.

One of the key elements that is at the base of a marketplace, industrial or of other type, is a platform. Smedlund and Faghankhani [31] define platform as *"Any physical or virtual space where different participants compose a market and a platform that participants orchestrate can be defined as a platform"*.

Platforms are typically composed of multi-layered structures, where the technological core element provides support for complementary technologies and software. Depending on the type of interactions between the participants, it is possible to identify different types of platforms such as: platforms that bring sellers and buyers together, platforms that help members of some group find a match in another group, platforms that measure transactions between participants, and platforms where participants share their input with other participants. Platforms can be classified as one-sided, two-sided, or multi-sided, depending on the number of participating groups. In n-sided platforms the users connect to each other, communicate and co-create value for themselves and for the other users and participants. The end-user must give something back to the platform, in order to benefit from the whole spectrum of platform functionalities.

Chen *et al.* in [76] defined online platforms, based on the definition of the European Commission, as *"digital platforms that enable consumers to find online information and businesses to exploit the advantages of e-commerce"*. They also classified online platforms into eight types according to user groups and business models: E-Commerce, Resource Sharing, Matching/Auction, Competitive Crowdsourcing, Non-competitive Crowdsourcing, Crowdfunding, Search Platforms, and Social Network Platforms. There are several factors for the success of online platforms such as the development costs, which are usually low for users because the main costs are associated with the development of the platform itself; the multi-sided revenue streams because on online platforms there is value created for all user groups involved; the online platforms have flexible scalability and can easily scale up operations; and online platforms are Widely accessible.

Nowadays there are two terms that often appear when one is talking about online business and that are often interchanged but that do have different meaning: e-commerce and e-marketplace. E-commerce is a business model where a given company buys products from several other companies and, through its web site, directly sells them to clients. E-marketplace is a different business model where the company that owns the web site only provides a platform, it does not directly buy or sell anything [77]–[80] (Figure 5).

Online businesses can be grouped into three big types of models as shown in Figure 6 [77], [81]:

- B2B - Business to Business is a type of transaction that is performed between businesses, such as a company buying materials or services needed for its production process, or re-selling products or services produced by others.
- B2C - Business to Consumer is when the transaction is between one business and its consumers who are the end-users of the product.
- C2C - Consumer to Consumer is a model where the transactions are executed directly between costumers.

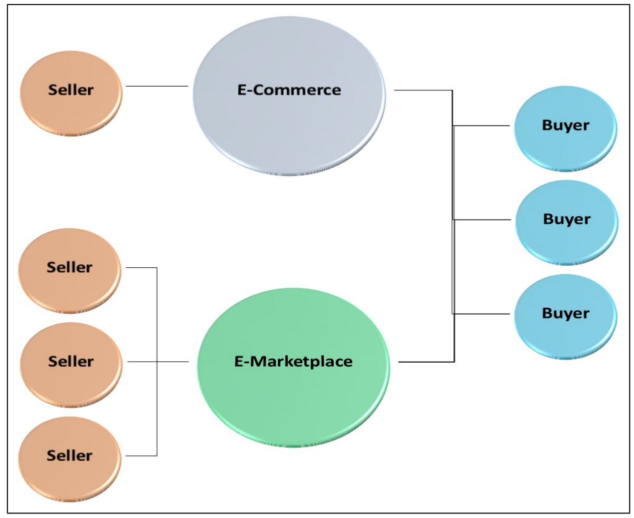


Figure 5. E-commerce vs E-marketplace

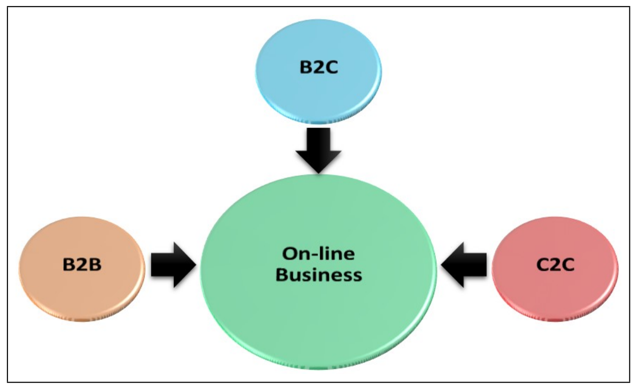


Figure 6. online Business Types

In order to properly define and establish the goals for the Cyber-Physical Industrial Marketplace concept, a set of requirements were collected and defined. These set of requirements allowed for the development of a workable marketplace, as well as provide a way of verifying the usefulness of the marketplace. With the intent of finding a valid and robust solution, it was necessary to validate all inherent functional and non-functional requirements, which in turn enables a full understanding of the customer needs and requirements. In order to satisfy the requirements of a real life application, some of the requirements were collected through a joint work with the Harms&Wende Karlsruhe – Germany Software Development team branch, which was part of the ReBorn project. The functional requirements collected are shown in Table I.

IV. Cyber-Physical Industrial Marketplace Concept

As mentioned before, in [1] the concept of a Marketplace for the cyber-physical production systems was presented. This concept emerged from the results of two European projects: the ReBorn project and the Selsus project.

Figure 7 presents what we consider to be the basic ideas that must be present in a Industrial Marketplace, specifically

TABLE I. Functional Requirements

Feature	User Management	Allow the creation of user accounts in a reliable and flexible way. User management functionalities such as user validation and customer data base management, by the platform administrator
	Access Control	Allow association of different roles to each user, which determines the type and amount of information available for that specific user
	Unlimited Products & Categories	Allow the unlimited creation of products and product categories
	Multi-Language	Support multiple languages on which information is displayed
	Gift Certificates	Allow gift certificates, which act as a key feature regarding platform growth, and inventory management
	Discount Coupons	Allow discount coupons to be generate at will, which is proven to be an important tool to product marketing
	Loyalty Program	Reward users which make multiple purchases on the marketplace, in order to increase the users commitment to the online platform
	Affiliates Program	Allow entities who are capable of selling and or promoting the marketplace products and services, or directing customers to it, in exchange for a percentage of a sale
	Search Optimization Tools	Search engine optimization in order to attempt to achieve the highest rank possible
	Product Reviews	Allow users to review products that are currently supported by the online platform
	Newsletters	Allow the platform administration the option to send bulk emails to existing customers, or customers that subscribed to a newsletter program
	Wish Lists	Offer the users the ability to create wish lists
Shipping	Shipping Filters	Display shipping costs according to product dimensions, weight, and delivery range of the items ordered
	Shipper Lookups	Offer a varied spectrum of courier delivery services regarding product shipping
Tax & Payment	Payment Options	Include built in support for payment gateways such as PayPal, and Authorize.net
	Tax Filters	Be able to take into account the appropriate tax rate applied to the customers country and state
Application Management	Industrial Equipment Software Update	Provide the latest application software releases upon request from each industrial equipment
	Full System Software Update	Allow, upon request by the client, a full system application software update up to the most recent and stable software version in a fully automated manner
	Specific Software Update	Allow individual update requests of each application
	Software Restore	Allow for software restore in case of any malfunction or defect is detected a regarding a particular software update
	Industrial Equipment Software Install	Allow for individual application installation as long as long as the application is compatible with the industrial equipment, and the industrial equipment owner possesses an available license for the to be installed application
	Industrial Equipment Software Uninstall	The industrial equipment upon uninstall of any application must notify the online platform

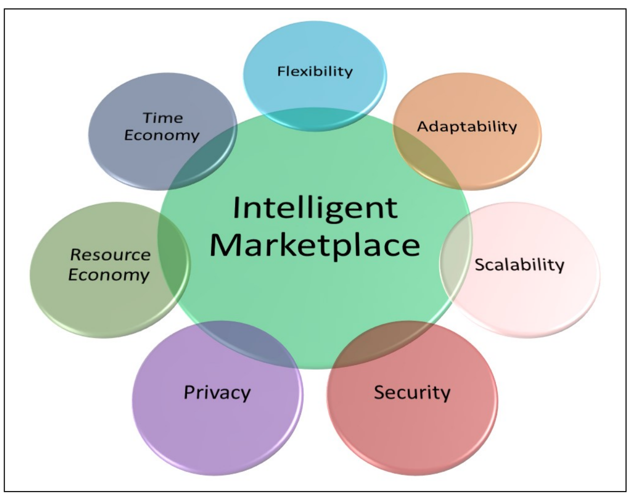


Figure 7. Proposed Solution Pillars

in a Cyber-Physical Industrial Marketplace. These basic ideas are the pillars of a Cyber-Physical Industrial Marketplace that is pointing toward the future:

- Flexibility - any future Industrial Marketplace must be flexible. In [82] flexibility is defined as ”the ease with which a system or component can be modified for use in applications or environments other than those for which it was specifically designed”. The marketplace must allow for new products or services exchange, and new protocols or technologies, through a seamless easy manner.
- Adaptability - is the way the marketplace internally adjust to flexibility. As time goes by and new technologies and applications emerge, an Industrial Marketplace must have the ability to adapt to new and different applications, through new technologies and protocols, without much effort.
- Scalability - any marketplace must be scalable, which means that it must be able to grow without losing functionality and flexibility. It must be able to deal with the natural increase, over time, of information and data flows.
- Security - is one of the most important aspects of these

type of businesses. For a marketplace to be intelligent it needs to be able to access a huge amount of information, ideally down to the equipment level. This introduces possible security breaches. Considering that in a marketplace many different kind of businesses are present with different kinds of security needs, the marketplace must be prepared to handle different security protocols.

- Privacy - walks hand in hand with security. For most companies, their information records are one of the most precious assets. This introduces the need for privacy and authentication. As in security, the marketplace must also be able to handle different schemes of privacy and authentication of users as well as of information.
- Resource Economy - in any business, small, medium, or large, there is a common goal: be able to save money to increase their profits. One way of accomplishing this is through a better resource management. Better resource management can be achieved through several ways, one of the most common is through cheaper materials or human force and by mass production of a single product. But the price of materials or workers can only decrease until a certain point, and although there are still many types of products that can be mass produce, customization is gaining strength. This means that new ways of resource economy must be found. The Cyber-Physical Industrial Marketplace can facilitate the development of these new ways such as new collaborations between different suppliers, reuse and re-purpose of machines, and creation of new services.
- Time Economy - another way of saving money in a business is by saving time. A well designed marketplace will allow for its users to save time through quickly supplying relevant information. In an Cyber-Physical Industrial Marketplace a user can find suppliers and partners easily and reach them with a simple click or message. This contributes to a more efficient and fast product design and production.

A. Proposed solution

With all the new technology that is virtually available everywhere to anyone, it is easy to imagine that everything can eventually be connected. This connectivity brings new information exchange channels, which leads to new business possibilities. Creating a marketplace that is more encompassing and with more functionalities than the current online stores, is the next natural and logical step.

The Cyber-Physical Industrial Marketplace, at first sight, works as a common online store. Users can register and buy products and sellers can also register and sell their products. The Marketplace is able to manage all the user and sellers accounts, as well as the shipping and payment services. This Cyber-Physical Industrial Marketplace is based on the ReBorn Marketplace [9], which is a multi-sided market, with service providers on one end and service consumers on the other.

As Figure 8 presents, the Cyber-Physical Industrial Marketplace is composed of several elements, such as sellers, equipment, services, buyers, platforms, and software. All these elements can have more than one role. For example, a buyer can be a seller or service provider, or a service provider can also provide software and equipment.

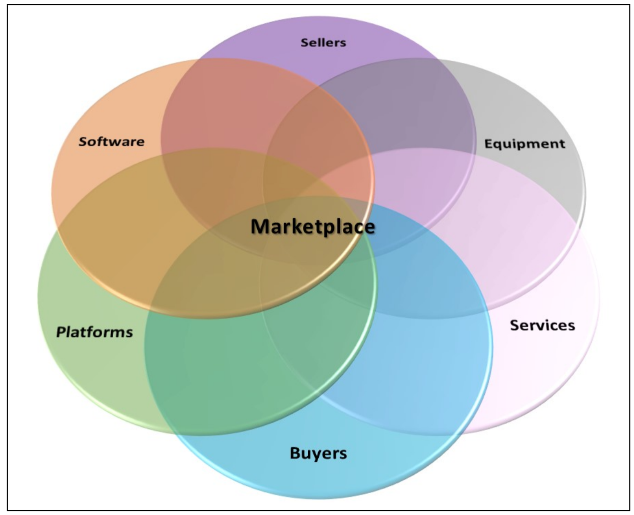


Figure 8. Cyber-Physical Industrial Marketplace Participants

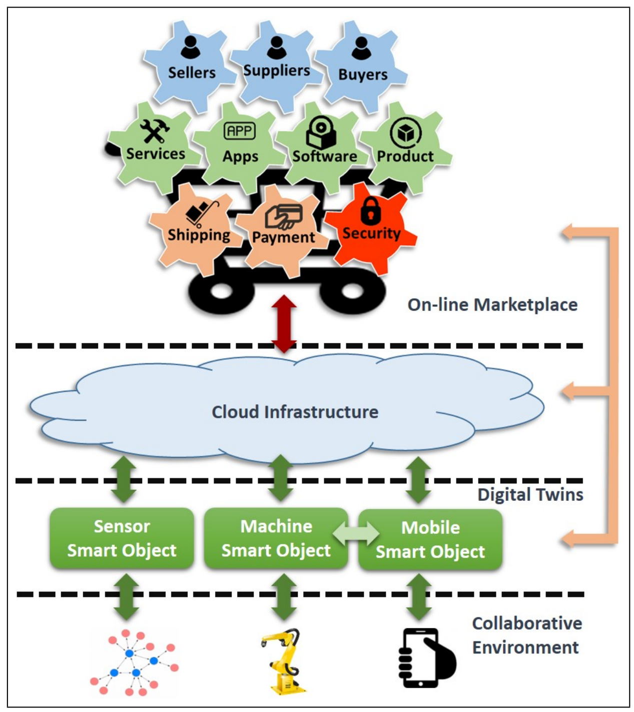


Figure 9. Cyber-Physical Industrial Marketplace Connection

Figure 9 provides a high level image of the proposed solution. As it can be seen, there are four main blocks in the proposed solution: the online marketplace, the cloud infrastructure, the digital twins, and the collaborative environment.

A key element in order to have an intelligent Cyber-Physical Industrial Marketplace is information. The bigger the amount of information available and the more precise the information collected is, the better and more accurate the management will be. Manufacturing equipment is becoming, itself, more intelligent and able of providing relevant and near real-time information. The problem is how to use this information, where to store it, and who or what has access to it. In the manufacturing world, information from the equipment is a desirable and often priceless asset. The right equipment configuration can be the difference between a top quality product and a medium or poor one. This has introduced a standstill in the widespread of industrial marketplaces. Another impasse on this type of business is that, although intelligent equipment is available, there is still a great amount of older equipment that has no sensors or any recent technology that allows gathering information.

In order to overcome these drawbacks the Smart Object concept was envisioned. The Smart Object can represent any element of the shop-floor such as machines, sensors, components, AGVs, smart-phones, and shop-floor operators with wearable sensor devices. The Smart Object is a digital representation of the shop-floor elements as shown in Figure 9, and it is based on the developments made in the ReBorn and Selsus projects. Both projects deepen and enhanced the smart component concept that was first developed in the European projects XPress [83], [84] and I-RAMP[3] [40], [85]. The main goal of these smart components was to enhance the equipment with new capabilities and functionalities. The Smart Object picks up on the results from these projects and takes it a step further. The idea is that the Smart Object can be treated as a black box. It can communicate with an equipment that already has sensors and is gathering information or it can have sensors added to it, in order to be used with older equipments that have no intelligence incorporated.

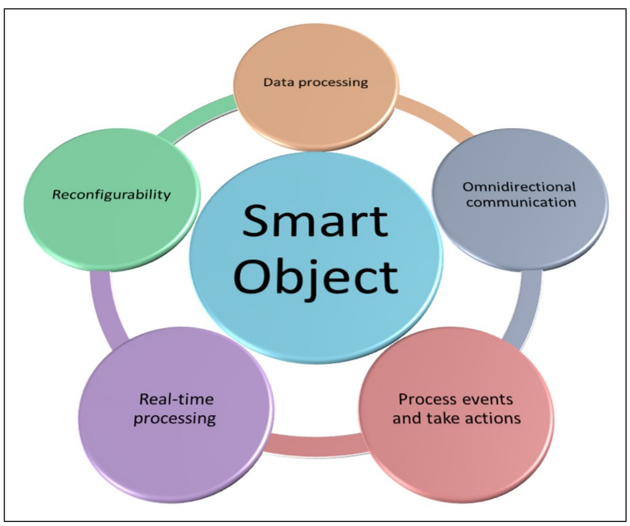


Figure 10. Smart Object Characteristics

The Smart Objects have five essential characteristics, which are presented in Figure 10 [17]. Smart Objects are reconfigurable because they are able of reconfiguring its internal operation in runtime and modular because they are able of extending its capabilities by adding new software modules, also at runtime. Smart Objects also have data processing capabilities, such as system state assessment and event and alarm detection. Another essential characteristic of the Smart Object is the omnidirectional communication and interface capabilities, which allow the system to communicate with devices at any level: lower level, such as sensors and machines; same level, such as other Smart Objects; or higher level, such as cloud servers and manufacturing systems. The Smart Objects are also able of processing events and of taking actions with a certain degree of intelligence and autonomy. The final characteristic is the ability for real-time acquisition, processing, and deliver of data collected.

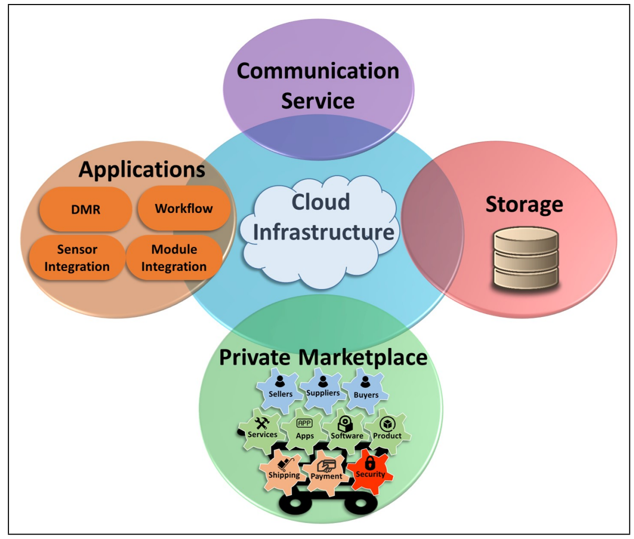


Figure 11. Cloud Infrastructure

The Smart Object can be considered as a bridge between the shop-floor and the virtual world. All the information collected and treated by the Smart Objects are sent to a cloud infrastructure. This infrastructure can be as simple or as complex as needed, depending on the type of business and the treatment that is required for the information collected. The Cloud Infrastructure is further detailed in Figure 11 and is composed by the following elements: communication service, storage, applications, and private marketplace.

The Communication Service is responsible for providing a way of exchanging information between the possible different platforms and software programs. This service has the capability of receiving and interpreting the requests to and from the Smart Objects, which allows all communications between the different levels of the system to be simple and clearly defined. In order for this communication the be easy and simple a semantic was defined [36]. There are two groups of possible communication: communication with the Cloud Infrastructure and communication with the Marketplace. In the Cloud Infrastructure, in order to directly communicate with the Smart Object three types of documents have been defined: the Smart Self Description (SSD), the Payload, and the Recipe Adjustment. These three documents are XML based. The SDD purpose is to allow the description of the shop-floor equipment. It has two main parts: meta information of the equipment, such as equipment manufacturer, serial number, equipment model, Smart Object name, and Smart Object unique identifier; and operational related data, such as which data needs to be collected and how, and to where it needs to be sent. The Payload document is has a very lightweight XML structure used to send operational process data to the Cloud Infrastructure. The Recipe Adjustment document is also a lightweight XML structure used to reconfigure the Smart Object processing process.

The Storage, as the name suggests, stores in form of a database, all the structural representation and operational data of the Smart Objects. This information will be accessible through the communication service and can be configured to determine what is stored, for how long, and who has permission to access it.

The Applications element encompasses a group of software applications that permits the easy interaction of the user with the system. These applications enable the user to control and reconfigure the system. Figure 11 presents some of the applications already developed: the Dynamic Modular Reconfiguration (DMR) [17], the Sensor Integration [35], the Workflow, and the Module Integrator [36]. The DMR purpose is to enable an easy and quick way of reconfiguring the Smart Object code responsible for processing data, which facilitates the reconfiguration and sensor integration in the system. The Sensor Integration is an application developed to aid the user to easily, though a graphical interface, define interpreters of the raw data packets. The purpose of this application is to allow the addition of new sensors by simply defining the type of message used and connecting the sensor on the shop-floor. The Workflow is also an application that intends to aid the user. The goal here is to allow the configuration of how data is process at a higher level, also through a graphical interface by means of a Directed Acyclic Graph (DAG) (Figure 12). The source nodes (red nodes) represent devices available that generate data, the intermediary nodes (blue nodes) are processing modules available to process data from source nodes, and the sink nodes (green nodes) that represent a higher level of processing. This allows to process multiple data sources and multiple Smart Objects, by simply drawing a DAG. The Module Integration application accede the integration of internal and external processing modules, which can be used in the DMR and in the Workflow.

The final element presented is the Private Marketplace. The idea behind the private marketplace emerged from the need of the industry to keep data private and local. The Private Marketplace is intended to be a copy of the online Marketplace, with all its features and services, but it is hosted on a private network. This Private Marketplace will act as a proxy between the online Marketplace and all the internal and private system elements. This allows for all the information collected by the Smart Objects to be sent and used by the Private Marketplace. This information can be used in applications such as maintenance planning or stock management. The Private Marketplace will allow to manage this information and facilitate the management of what can be uploaded into the online Marketplace. On the other hand, any software update or new feature that becomes available on the online Marketplace, can be downloaded to the Private Marketplace, through direct connection or manual update, and then be used to update the Smart Objects as needed.

Besides the three documents, described above, used by the Cloud Infrastructure to communicate with the Smart Object, there is also the need to communicate with the Marketplace, in order to have the ability to update the Smart Object software or to add new features, such as a security module or a new data processing algorithm. The communication with the Marketplace is performed through a RESTful Web Services,

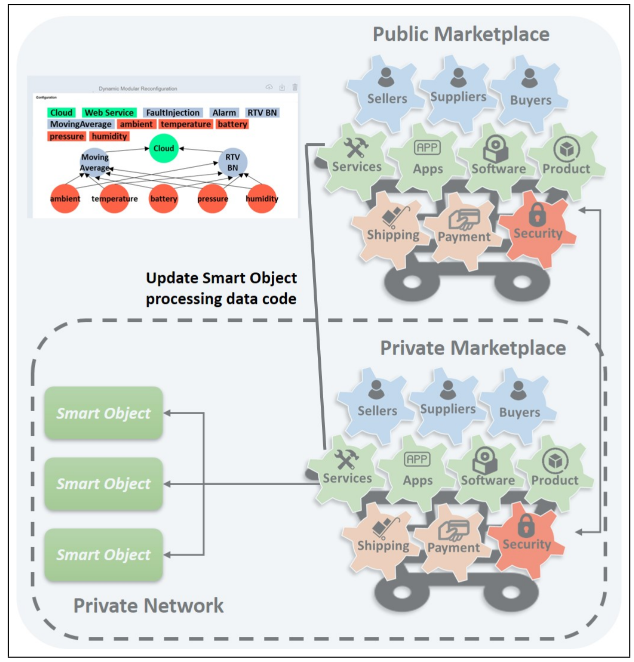


Figure 12. Dynamic Modular Reconfiguration Application

which is a Representational State Transfer (REST) web service based architecture. The REST was chosen because it is a web standard based architecture that uses the HTTP Protocol for data communication. In the REST architecture, every component is a resource and can be accessed by a common interface using HTTP standard methods, such as GET (a read only access to a resource), PUT (to create a new resource), DELETE (to remove a resource), POST (to update an existing resource or create a new resource), and OPTIONS (to get the supported operations on a resource). From the possible text representations that are supported the JSON and XML where chosen.

The online Marketplace, as shown in Figure 9, provides several services and functionalities. There are several functionalities that are inherent to an online marketplace. These inherent functionalities include all management related to users, products, shipping, and payment as presented in Table I. The management activity encompasses operations that include adding, deleted, or changing information from users; adding, deleting, changing information from products, and keeping track of the products stocks; keeping track of shipment activities; and manage all payments related activities. But aside from all these regular activities, in order to be considered intelligent and a Cyber-physical Industrial Marketplace, it must provide other services that can aid users on their activities. For example alert the user for equipments that are similar to what the user is searching or provide updates on new software or new complementary equipment compatible with what that the user has acquired.

A key functionality for a Cyber-Physical Industrial Marketplace is the ability to manage the shop-floor equipment. As mentioned before, a direct connection between the shop-floor and the internet, is not desirable and most of the times not even possible. This restriction was behind the concept of the Private Marketplace. As mentioned before the Private Marketplace is a copy of the online Marketplace, which can execute all the services, in a private setting with no online connection. But this Private Marketplace can also synchronize with the online Marketplace all the information collected by the Smart Objects that has been defined as not private, and also receive updates from the online Marketplace.

One important feature is the Industrial Software Manager. This manager is responsible for the management of the application content of each Smart Object. This allows the upgrade of the Smart Object software, in an easy and simple way, facilitating the integration of new technologies and functionalities in the industrial equipment on the fly.

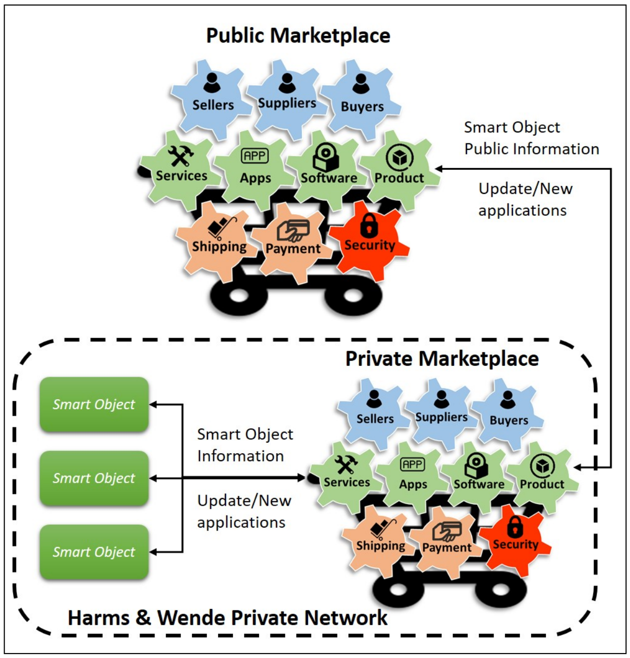


Figure 13. Practical example

The implementation of the Private/Public Marketplace concept was tested at the Harms&Wende (HWH) Karlsruhe [86], as part of the joint work collaboration within the ReBorn project. HWH is a component supplier specialized in welding control systems for various sectors, offering resistance welding equipment in form of control devices as well as quality assurance systems. HWH customers are often welding machines manufacturers who use the HWH welding control units.

Figure 13 provides an overview of the implementation of the Private/Public Marketplace concept presented. As it is shown, there are three main components: the Smart Objects, the Private Marketplace and the Public Marketplace. For the test performed at the HWH, the Smart Object was implemented on a dedicated hardware platform, a RaspBerry Pi SBC, to enable an easy implementation and test of the concept. Inside the Marketplace, as previously stated, there is the Industrial Software Manager, which, in the case of the Private Marketplace, locally manages the application content of each equipment's Smart Object, while the equipment is installed in a production line. This Industrial Software Manager can receive requests and update the Smart Object software, through a REST based web service [9]. The Industrial Software Manager in the Private Marketplace can synchronize its information as well as receive updates from the Industrial Software Manager in the Public Marketplace.

Another important and crucial feature is the Security Manager. This manager facilitates the possibility of having a security layer over all the communications that occur within the system, from the Smart Object up to the online Marketplace. This manager will provide privacy and authentication modules, that can be configured to use different algorithms, and which can be added to every element of the system.

The purpose of the Maintenance Manager, as can be inferred from the name, is to aid in the maintenance planning. Using predictive algorithms based on the information collected by the Smart Objects, the Maintenance Manager can aid in the scheduling of the equipment's maintenance, in order to minimize the production down time. But this manager can also be useful in a different scenario. Consider an equipment supplier that wants to increase their business by adding maintenance services to its portfolio. Many equipment suppliers sell across the country and even to across countries. The Maintenance Manager, based on the information collected by the Smart Objects, can provide a maintenance or inspection schedule that takes into account the geographical placing of the equipment as well as the current state of the equipment. This will allow the equipment supplier to minimize the travelling costs and also provide a more active preventive maintenance service. The Maintenance Manager will generate possible maintenance routes, based on configurable variables, such as minimize costs or travel time.

As mentioned before, one key functionality of this type of systems is to be able to change, on the fly, what is being processed in the Smart Objects. With that in mind, the Simulation Dashboard was defined. As the Maintenance Manager, this Simulation Dashboard can be used in two distinct scenarios. It can be used privately by a company that needs a new functionality for its Smart Objects or it can be used by developers that want to sell new modules for the Smart Objects on the Marketplace. The Simulation Dashboard facilitates the design, implementation, and test of new features, such as new metric calculations or a new form of processing raw data in the Smart Objects. After the user implements and tests, through simulation the new functionality developed, it can be sent to the Smart Objects or stored in the online Marketplace to be sold.

B. Advantages and disadvantages of a Cyber-Physical Industrial Marketplace

With the digital revolution, business models have changed and evolved. Online Marketplaces are growing at a fast pace. They provide a way for companies to quickly and easily find new costumers and expand their business. They also provide a way for new opportunities to establish partnerships between traders and suppliers. Using online Marketplaces reduces marketing costs compared to other sales channels. It also allows for new opportunities for overseas sales, without the costs of setting up stores across different countries. Being able to operate a round-the-clock business reduces time constraints and problems with international trading hours.

For buyers, online Marketplaces are also a quick way to find companies, design and produce prototypes, and buy products easily. Online Marketplaces provide a convenient way to compare prices and products from a single source. Using a credible online Marketplace contributes to the building of a greater level of trust between the sellers and the buyers. It also improves the level of transparency, through the availability of prices and stock levels in an open environment.

The downfall of online Marketplaces is thrust or more exactly the lack of trust. The online Marketplaces can provided a venue for bad players to set up store-fronts and traffic in damaged, counterfeit, or falsely products. Another issue also related with trust is the management of the information that can be exchanged, how can privacy and security be guaranteed. Credit card fraud is also a huge concern when dealing with online Marketplaces businesses.

V. Conclusion

Industry 4.0 has brought to light new concepts and technologies. Cyber-Physical Systems, the Internet of Things, and Cloud computing pave the way for the Smart Factories concept. Within the Smart Factories approach, a large amount of data is available to be used. All the available data can be processed using data analytic algorithms, which can then be used to improve productivity and production planning as well as allow for a better equipment predictive maintenance. In a world where competition is global, the industry must be able to respond to unpredictable and rapid market changes. Manufacturing systems must become flexible, easily upgradable, and allow for new technologies and functions to be readily integrated. This creates the need for novel manufacturing control systems able to cope with the increased complexity required to manage product and production variability in mass customized manufacturing.

A solution for this challenge is through the Smart Components concept, a logical encapsulation of the manufacturing industrial equipment. To organise and store information of industrial machines, not only for potential customers to check and compare different devices, but also to provide an interface that allows for automatic update of the industrial equipment technologies and functionalities on the fly, a Cyber-Physical Industrial Marketplace is key.

In this paper an architecture for a Cyber-Physical Industrial Marketplace is proposed. This work is the result of combining some of the results and experience collected over two European projects: the ReBorn and the Selsus project. Within these two projects, the contributions produced, which are also at the basis of this proposed architecture are the ReBorn Marketplace, the SelSus Dashboard, and the combination of the functionalities of the Smart Components implemented on both projects.

The ultimate goal of the proposed architecture is to provide an efficient and easy to use Industrial Marketplace that encompasses all the activities related to the manufacturing field. The objective is to provide a route for the flow of information from the equipment at the shop-floor, up to the Marketplace and back. This proposed Cyber-Physical Industrial Marketplace will not only provide a virtual shop for equipment but also for smart components, software, and services. In order to be truly intelligent, this marketplace must also be flexible and configurable in order to adjust to each user needs. The proposed Cyber-Physical Industrial Marketplace allows to take advantage of all the current technological advances, and contributes for a safe and reliable way of using all the available information.

References

[1] S. Aguiar, R. Pinto, J. Reis, and G. Gonçalves, "A marketplace for cyber-physical production systems: Architecture and key enablers," in INTELLI 2017, The Sixth International Conference on Intelligent Systems and Applications. IARIA, 2017, pp. 81–86.

[2] K. Henning, "Recommendations for implementing the strategic initiative industrie 4.0," 2013.

[3] M. Rüßmann, M. Lorenz, P. Gerbert, M. Waldner, J. Justus, P. Engel, and M. Harnisch, "Industry 4.0: The future of productivity and growth in manufacturing industries," 2015.

[4] S. Weyer, M. Schmitt, M. Ohmer, and D. Gorecky, "Towards industry 4.0-standardization as the crucial challenge for highly modular, multi-vendor production systems," in Ifac - Papers online, vol. 48, no. 3. Elsevier, 2015, pp. 579–584.

[5] H. Lasi, P. Fettke, H.-G. Kemper, T. Feld, and M. Hoffmann, "Industry 4.0," Business & Information Systems Engineering, vol. 6, no. 4, 2014, pp. 239–242.

[6] M. Wollschlaeger, T. Sauter, and J. Jasperneite, "The future of industrial communication: Automation networks in the era of the internet of things and industry 4.0," IEEE Industrial Electronics Magazine, vol. 11, no. 1, 2017, pp. 17–27.

[7] Y. Liao, F. Deschamps, E. d. F. R. Loures, and L. F. P. Ramos, "Past, present and future of industry 4.0-a systematic literature review and research agenda proposal," International Journal of Production Research, vol. 55, no. 12, 2017, pp. 3609–3629.

[8] E. Hofmann and M. Rüsch, "Industry 4.0 and the current status as well as future prospects on logistics," Computers in Industry, vol. 89, 2017, pp. 23–34.

[9] R. Fonseca, S. Aguiar, M. Peschl, and G. Gonçalves, "The reborn marketplace: an application store for industrial smart components," in INTELLI 2016, The Fifth International Conference on Intelligent Systems and Applications. IARIA, 2016, pp. 136–141.

[10] F. Zezulka, P. Marcon, I. Vesely, and O. Sajdl, "Industry 4.0–an introduction in the phenomenon," IFAC-PapersOnLine, vol. 49, no. 25, 2016, pp. 8–12.

[11] Y. Lu, "Industry 4.0: A survey on technologies, applications and open research issues," Journal of Industrial Information Integration, 2017.

[12] X. Li, D. Li, J. Wan, A. V. Vasilakos, C.-F. Lai, and S. Wang, "A review of industrial wireless networks in the context of industry 4.0," Wireless networks, vol. 23, no. 1, 2017, pp. 23–41.

[13] S. Wang, J. Wan, D. Li, and C. Zhang, "Implementing smart factory of industrie 4.0: an outlook," International Journal of Distributed Sensor Networks, vol. 12, no. 1, 2016, p. 3159805.

[14] Y. Koren, U. Heisel, F. Jovane, T. Moriwaki, G. Pritschow, G. Ulsoy, and H. Van Brussel, "Reconfigurable manufacturing systems," CIRP Annals-Manufacturing Technology, vol. 48, no. 2, 1999, pp. 527–540.

[15] M. G. Mehrabi, A. G. Ulsoy, Y. Koren, and P. Heytler, "Trends and perspectives in flexible and reconfigurable manufacturing systems," Journal of Intelligent manufacturing, vol. 13, no. 2, 2002, pp. 135–146.

[16] "Review of standardization opportunities in smart industrial components," URL: http://publica.fraunhofer.de/starweb/servlet.starweb?path=epub.web&search=N-413239 [accessed: 2017-11-27].

[17] L. Neto, J. Reis, R. Silva, and G. Gonçalves, "Sensor selcomp, a smart component for the industrial sensor cloud of the future," in Industrial Technology (ICIT), 2017 IEEE International Conference on. IEEE, 2017, pp. 1256–1261.

[18] D. Wu, D. W. Rosen, L. Wang, and D. Schaefer, "Cloud-based design and manufacturing: A new paradigm in digital manufacturing and design innovation," Computer-Aided Design, vol. 59, 2015, pp. 1–14.

[19] X. Xu, "From cloud computing to cloud manufacturing," Robotics and computer-integrated manufacturing, vol. 28, no. 1, 2012, pp. 75–86.

[20] P. Wang, R. X. Gao, and Z. Fan, "Cloud computing for cloud manufacturing: benefits and limitations," Journal of Manufacturing Science and Engineering, vol. 137, no. 4, 2015, p. 040901.

[21] F. Tao, L. Zhang, V. Venkatesh, Y. Luo, and Y. Cheng, "Cloud manufacturing: a computing and service-oriented manufacturing model," Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture, vol. 225, no. 10, 2011, pp. 1969–1976.

[22] X. V. Wang and X. W. Xu, "An interoperable solution for cloud manufacturing," Robotics and Computer-Integrated Manufacturing, vol. 29, no. 4, 2013, pp. 232–247.

[23] L. M. Kaufman, "Data security in the world of cloud computing," IEEE Security & Privacy, vol. 7, no. 4, 2009.

[24] Y. Jadeja and K. Modi, "Cloud computing-concepts, architecture and challenges," in Computing, Electronics and Electrical Technologies (ICCEET), 2012 International Conference on. IEEE, 2012, pp. 877–880.

[25] P. Mell, T. Grance et al., "The nist definition of cloud computing," 2011.

[26] E. Luoma, M. Rönkkö, and P. Tyrväinen, "Current software-as-a-service business models: Evidence from finland," in International Conference of Software Business. Springer, 2012, pp. 181–194.

[27] M. Sääksjärvi, A. Lassila, and H. Nordström, "Evaluating the software as a service business model: From cpu time-sharing to online innovation sharing," in IADIS international conference e-society. Qawra, Malta, 2005, pp. 177–186.

[28] L. M. Vaquero, L. Rodero-Merino, J. Caceres, and M. Lindner, "A break in the clouds: towards a cloud definition," ACM SIGCOMM Computer Communication Review, vol. 39, no. 1, 2008, pp. 50–55.

[29] A. Hagiu and J. Wright, "Multi-sided platforms," International Journal of Industrial Organization, vol. 43, 2015, pp. 162–174.

[30] M. Cusumano, "Technology strategy and management the evolution of platform thinking," Communications of the ACM, vol. 53, no. 1, 2010, pp. 32–34.

[31] A. Smedlund and H. Faghankhani, "Platform orchestration for efficiency, development, and innovation," in System Sciences (HICSS), 2015 48th Hawaii International Conference on. IEEE, 2015, pp. 1380–1388.

[32] "ReBorn Project web site," URL: http://www.reborn-eu-project.org/ [accessed: 2017-11-27].

[33] S. Aguiar, R. Pinto, J. Reis, and G. Gonçalves, "Life-cycle approach to extend equipment re-use in flexible manufacturing," in INTELLI 2016, The Fifth International Conference on Intelligent Systems and Applications. IARIA, 2016, pp. 148–153.

[34] "SelSus Project web site," URL: http://www.selsus.eu/ [accessed: 2017-11-27].

[35] R. Silva, J. Reis, L. Neto, and G. Gonçalves, "Universal parser for wireless sensor networks in industrial cyber physical production systems," in Industrial Informatics (INDIN), 2017 IEEE 15th International Conference on. IEEE, 2017, pp. 633–638.

[36] J. Reis, R. Pinto, and G. Gonçalves, "Human-centered application using cyber-physical production system," in 43rd Annual Conference of the IEEE Industrial Electronics Society (IECON). IEEE, 2017.

[37] J. Reis, "SelSus - White paper on sensor cloud," URL: http://www.selsus.eu/fileadmin/mount/documents/SelSus - D3.5 - White paper on sensor clouds.pdf [accessed: 2018-01-03].

[38] F. Banaie and S. A. H. Seno, "A cloud-based architecture for secure and reliable service provisioning in wireless sensor network," in Computer and Knowledge Engineering (ICCKE), 2014 4th International eConference on. IEEE, 2014, pp. 96–101.

[39] K. R. Llanes, M. A. Casanova, and N. M. Lemus, "From sensor data streams to linked streaming data: a survey of main approaches," Journal of Information and Data Management, vol. 7, no. 2, 2017, p. 130.

[40] G. Gonçalves, J. Reis, R. Pinto, M. Alves, and J. Correia, "A step forward on intelligent factories: A smart sensor-oriented approach," in Emerging Technology and Factory Automation (ETFA), 2014 IEEE. IEEE, 2014, pp. 1–8.

[41] W. Shen, Q. Hao, H. J. Yoon, and D. H. Norrie, "Applications of agent-based systems in intelligent manufacturing: An updated review," Advanced engineering INFORMATICS, vol. 20, no. 4, 2006, pp. 415–431.

[42] "Sensor ML," URL: http://www.ogcnetwork.net/SensorML [accessed: 2017-01-01].

[43] "Open Geospatial Consortium," URL: http://www.opengeospatial.org/ [accessed: 2018-01-15].

[44] J. Shneidman, P. Pietzuch, J. Ledlie, M. Roussopoulos, M. Seltzer, and M. Welsh, "Hourglass: An infrastructure for connecting sensor networks and applications," Tech. Rep., 2004.

[45] M. Yuriyama and T. Kushida, "Sensor-cloud infrastructure-physical sensor management with virtualized sensors on cloud computing," in Network-Based Information Systems (NBiS), 2010 13th International Conference on. IEEE, 2010, pp. 1–8.

[46] J. Yan, Y. Ma, L. Wang, K.-K. R. Choo, and W. Jie, "A cloud-based remote sensing data production system," Future Generation Computer Systems, 2017.

[47] C. Yang, S. Lan, W. Shen, G. Q. Huang, X. Wang, and T. Lin, "Towards product customization and personalization in iot-enabled cloud manufacturing," Cluster Computing, 2017, pp. 1–14.

[48] L. Zhang, Y. Luo, F. Tao, B. H. Li, L. Ren, X. Zhang, H. Guo, Y. Cheng, A. Hu, and Y. Liu, "Cloud manufacturing: a new manufacturing paradigm," Enterprise Information Systems, vol. 8, no. 2, 2014, pp. 167–187.

[49] K. M. Alam and A. El Saddik, "C2ps: A digital twin architecture reference model for the cloud-based cyber-physical systems," IEEE Access, vol. 5, 2017, pp. 2050–2062.

[50] L. Neto, J. Reis, D. Guimarães, and G. Gonçalves, "Sensor cloud: Smartcomponent framework for reconfigurable diagnostics in intelligent manufacturing environments," in Industrial Informatics (INDIN), 2015 IEEE 13th International Conference on. IEEE, 2015, pp. 1706–1711.

[51] Y. Chiang and D. Lee, "Smart manufacturing with the internet of makers," Journal of the Chinese Institute of Engineers, vol. 40, no. 7, 2017, pp. 585–592.

[52] A. Alamri, W. S. Ansari, M. M. Hassan, M. S. Hossain, A. Alelaiwi, and M. A. Hossain, "A survey on sensor-cloud: architecture, applications, and approaches," International Journal of Distributed Sensor Networks, vol. 9, no. 2, 2013, p. 917923.

[53] J. I. R. Molano, J. M. C. Lovelle, C. E. Montenegro, J. J. R. Granados, and R. G. Crespo, "Metamodel for integration of internet of things, social networks, the cloud and industry 4.0," Journal of Ambient Intelligence and Humanized Computing, 2017, pp. 1–15.

[54] C. Perera, C. H. Liu, and S. Jayawardena, "The emerging internet of things marketplace from an industrial perspective: A survey," IEEE Transactions on Emerging Topics in Computing, vol. 3, no. 4, 2015, pp. 585–598.

[55] C. C. Albrecht, D. L. Dean, and J. V. Hansen, "Marketplace and technology standards for b2b e-commerce: progress, challenges, and the state of the art," Information & Management, vol. 42, no. 6, 2005, pp. 865–875.

[56] B. Varghese, N. Wang, D. S. Nikolopoulos, and R. Buyya, "Feasibility of fog computing," arXiv preprint arXiv:1701.05451, 2017.

[57] M. Satyanarayanan, "The emergence of edge computing," Computer, vol. 50, no. 1, 2017, pp. 30–39.

[58] R. Roman, P. Najera, and J. Lopez, "Securing the internet of things," Computer, vol. 44, no. 9, 2011, pp. 51–58.

[59] Q. Jing, A. V. Vasilakos, J. Wan, J. Lu, and D. Qiu, "Security of the internet of things: perspectives and challenges," Wireless Networks, vol. 20, no. 8, 2014, pp. 2481–2501.

[60] A. Sajid, H. Abbas, and K. Saleem, "Cloud-assisted iot-based scada systems security: A review of the state of the art and future challenges," IEEE Access, vol. 4, 2016, pp. 1375–1384.

[61] S. Sicari, A. Rizzardi, L. A. Grieco, and A. Coen-Porisini, "Security, privacy and trust in internet of things: The road ahead," Computer Networks, vol. 76, 2015, pp. 146–164.

[62] X. Lu, Q. Li, Z. Qu, and P. Hui, "Privacy information security classification study in internet of things," in Identification, Information and Knowledge in the Internet of Things (IIKI), 2014 International Conference on. IEEE, 2014, pp. 162–165.

[63] A. Alcaide, E. Palomar, J. Montero-Castillo, and A. Ribagorda, "Anonymous authentication for privacy-preserving iot target-driven applications," Computers & Security, vol. 37, 2013, pp. 111–123.

[64] A. Puliafito, A. Celesti, M. Villari, and M. Fazio, "Towards the integration between iot and cloud computing: an approach for the secure self-configuration of embedded devices," International Journal of Distributed Sensor Networks, 2015.

[65] J.-X. Hu, C.-L. Chen, C.-L. Fan, and K.-h. Wang, "An intelligent and secure health monitoring scheme using iot sensor based on cloud computing," Journal of Sensors, vol. 2017, 2017.

[66] J. M. Taylor and H. R. Sharif, "Security challenges and methods for protecting critical infrastructure cyber-physical systems," in Selected Topics in Mobile and Wireless Networking (MoWNeT), 2017 International Conference on. IEEE, 2017, pp. 1–6.

[67] A. Zahra, M. Nizamuddin, and Z. Jaffery, "Implementation & analysis of security protocols for wireless sensor network," International Journal of Electronics Engineering, vol. 2, no. 1, 2010, pp. 111–113.

[68] "Material Handling Solutions," URL: http://www.materialhandlingpub.com [accessed: 2018-01-18].

[69] "Industrial Auctions and Liquidations," URL: https://www.aucto.com/ [accessed: 2018-01-18].

[70] "Global Industrial Online Marketplace," URL: http://www.materialhandlingpub.com/global-industrial-marketplace/ [accessed: 2018-01-16].

[71] "Industrial Market Place," URL: http://www.impmagazine.com/ [accessed: 2018-01-16].

[72] "Global Equipment Company Inc." URL: http://www.globalindustrial.com/ [accessed: 2018-01-18].

[73] "Intelligent Plant," URL: https://www.intelligentplant.com/ [accessed: 2018-01-18].

[74] "Advantech," URL: http://wise-paas.advantech.com/en-us/marketplace [accessed: 2018-01-18].

[75] B. J. Corbitt, T. Thanasankit, and H. Yi, "Trust and e-commerce: a study of consumer perceptions," Electronic commerce research and applications, vol. 2, no. 3, 2003, pp. 203–215.

[76] Y.-J. Chen, T. Dai, C. G. Korpeoglu, E. Körpeoğlu, O. Sahin, C. S. Tang, and S. Xiao, "Innovative online platforms: Research opportunities," 2018.

[77] V. Kumar and E. G. Raheja, "Business to business (b2b) and business to consumer (b2c) management," International Journal of Computers & Technology, vol. 3, no. 3b, 2012, pp. 447–451.

[78] W. Raisch, The eMarketplace: Strategies for Success in B2B eCommerce. New York, NY, USA: McGraw-Hill, Inc., 2002.

[79] D. Chaffey, E-business and E-commerce Management: Strategy, Implementation and Practice. Pearson Education, 2007.

[80] R. T. Wigand, "Electronic commerce: Definition, theory, and context," The information society, vol. 13, no. 1, 1997, pp. 1–16.

[81] S. Iankova, I. Davies, C. Archer-Brown, B. Marder, and A. Yau, "A comparison of social media marketing between b2b, b2c and mixed business models," Industrial Marketing Management, 2018.

[82] I. ISO, "Ieee, systems and software engineering–vocabulary," IEEE computer society, Piscataway, NJ, 2010.

[83] "XPress Project web site," URL: https://www.steinbeis-europa.de/en/sectors-projects/advanced-manufacturing-technologies/xpress.html [accessed: 2017-11-27].

[84] F. Almeida, P. Dias, G. Gonçalves, M. Peschl, and M. Hoffmeister, "A proposition of a manufactronic network approach for intelligent and flexible manufacturing systems," International Journal of Industrial Engineering Computations, vol. 2, no. 4, 2011, pp. 873–890.

[85] "IRAMP Project web site," URL: http://www.i-ramp3.eu/ [accessed: 2017-11-27].

[86] "Harms & Wende Karslruhe," URL: https://www.harms-wende.de/en/others/contact/hwh-karlsruhe/ [accessed: 2018-01-18].

Smartphone-based Data Collection with Stunner, the Reality of Peer-to-Peer Connectivity and Web Real-Time Communications Using Crowdsourcing: Lessons Learnt while Cleaning the Data

Zoltán Szabó*, Vilmos Bilicki*, Árpád Berta†, and Zoltán Richárd Jánki*
*Department of Software Engineering
University of Szeged, Hungary
Email: {szaboz, bilickiv, jankiz}@inf.u-szeged.hu
†MTA-SZTE Research Group on AI
University of Szeged, Hungary
Email: berta@inf.u-szeged.hu

Abstract—**The increasing popularity of smartphones makes them popular tools for various big data collecting crowdsourcing campaigns, but there are still many open questions about the proper methodology of these campaigns. Beyond this, despite the growing popularity of this type of research, there are familiar difficulties and challenges in handling a wide range of uploads, maintaining the quality of the datasets, cleaning the datasets containing noisy, incorrect data, motivating the participants, and providing support for data collection regardless of the remoteness of the device. In order to collect information about the Network Address Translation (NAT)-related environment and the Peer-to-Peer (P2P) networking capabilities of mobile phones, we utilized a crowdsourcing approach. We collected more than 70 million data records from over 100 countries measuring the NAT characteristics of more than 1,300 carriers and over 35,000 WiFi environments during the three-year project. Since then, we have also expanded and released our application to collect even more data concerning the Peer-to-Peer capabilities. Here, we introduce our data collecting and Peer-to-Peer architectures, some of the most prominent problems we have encountered since its launch, some of the solutions and proposed solutions to handle difficulties.**

Keywords–smartphones; data cleaning; Peer-to-Peer; crowdsourcing.

I. Introduction

In recent years, smartphones have become part of our everyday lives. Their wide range of uses along with multiple sensors, networking and computational capabilities have also made them seemingly ideal platforms for research. One research area is data collection, with the collected datasets available for a wide area of analysis, including network mapping, discovering and analysing various networks, and the network coverage of certain areas.

Different research teams from all over the world have discovered these new opportunities, and they employ smartphones as crowdsourcing tools in a wide variety of ways. Through crowdsourcing, they assign tasks to different users with different device types to collect data in real-life situations, or a monitored environment, providing huge amounts of realistic data. In recent years, we have seen a lot of successful, and interesting approaches to this methodology. In our conference paper, we have shown our approach to data collection and data cleaning [1].

However, there is still a question of how exactly crowdsourcing campaigns should be implemented. Several research projects, such as SmartLab [2], the behaviour-based malware detection system Crowdroid [3], and the cross-space public information crowdsensing system FlierMeet [4] recruited a small number of users, who could be trusted, contacted if necessary, and provided the data taken from a known environment, specifically chosen, or created for the crowdsourcing project. This limited the variability and the amount of the data, but the results were of a high quality and easy to validate.

Another approach for recruitment is to upload the smartphone application to the Google Play store, or the Apple App Store, making it available for download by anyone world-wide, and opening up data collecting opportunities for anyone who agreed to the terms and services of the software package. With proper marketing, the results could include enormous datasets obtained from around the world. The NoiseTube project [5] for crowdsourcing noise pollution detection was downloaded by over 500 people from over 400 regions world-wide. The Dialäkt App [6], one of the most well-known crowdsourcing campaigns in recent years, was the most downloaded iPhone app in Switzerland after its launch, with wide media coverage, and over 78,000 downloads from 58,923 users by the time they had published their results. Many more datasets were collected in the Bredbandskollen project, later to be used in various smartphone-based research projects [7], which has collected network data from 3,000 different devices and over 120 million records since its launch in 2007, and the OpenSignal [8] application, which between 2012 and 2013 collected over 220 million data records from more than 530,000 devices and from over 200 countries.

However, collecting data using smartphones is not without its difficulties, and there are a number of challenges when smartphones are used as the prime source of information. Among these, battery consumption and network state are among the most important elements, as constantly accessing the state of the phone sensors and listening to specific events takes a heavy toll on the battery, making data collection inadvisable in certain situations (for example, after a device signalled a battery low event), and it is feasible, but pointless in other situations (the phone is on a charger while the user is asleep - the energy is there, but the valuable information is only a fraction of what we would get from an active user). Network state again has to be taken into account, as even today in many environments, we cannot ensure that a device will always have a connection strong enough to send the collected data to the server. Privacy is also an issue, since the data has to remain identifiable yet not contain any trace of personal information.

Aside from all of the above, if data collection was successful, we still have the problem of noisy, incorrect, disorganised data. We have to take into consideration the fact that there are different devices, different versions of the same OS, bugs, such as duplicated records uploaded by the client on network error and damaged records resulting from a similar event. We also have to take user interference into account, who may not wish to provide valuable data (e.g., by deliberately leaving the phone at home on a charger or having it switched off during specific hours, etc.). Their results will still be counted as valuable data, but this can severely distort the collected dataset, as well as the results used in evaluations.

An even bigger problem, when crowdsourcing is a global campaign, is that of time synchronisation. Not only do we have to find a good solution for the various time zones of the devices, but also the different time codes of the phone collecting and sending the data and the server storing this data, and recognize the possibility that the user might have manually altered the date and time on the test phone as well.

All of the possibilities mentioned above result in a mixed situation where the power and potential of the smartphones, as research tools cannot be denied, but to acquire correct, useful data is a challenge in itself. This requires careful planning, taking into account almost every possible cause of data distortion, well-defined filters and data cleaning algorithms before any actual research can be performed on the data collected. In this article, we are going to present our solutions with this type of data collection, and our solutions to the problems that emerged.

Our goal was to develop an Android app in order to collect important network information for research on the P2P capabilities of smartphones, including the NAT type, network type and network provider. It does so by taking measurements on a regular basis as well as during specific events. When taking the measurements, the app sends a request to a randomly chosen Session Traversal Utilities for NAT (STUN) Server from a list, displaying useful network information, such as the IP address and NAT type to the user, while also storing the necessary data in an SQLite database, which later gets uploaded to a data collector server for analysis. The application called Stunner has been available for download from the Play Store since December 2013 [9].

Since then we have continued the development, and the current version of the application also contains the implementation of the WebRTC project. Soon, we aim to implement P2P algorithms through P2P connections and collect the specific P2P metrics of the results. In our latest developments we also upgraded the data storage, along with the classic data collector server, both the NAT measurements and the new P2P data are stored on a Parse database server. The new version of Stunner is available in the Play Store since 25th April, 2018.

Using the collected data, we can compare the earlier NAT measurements to the new P2P measurements. Also, in the future, we will be able to define the graph model of a worldwide, P2P smartphone network. In this model, we seek to test various P2P protocols to measure the capabilities of a serverless network architecture, where the phones can slowly update their datasets and generate various statistics, without the data ever leaving this smartphone network. The ultimate goal of our research is the creation of an Application Programming Interface (API), through which developers can utilize these P2P capabilities to create various data collecting and processing applications (for example, general mood or health statistic researching applications for a specific region) without the need of a processing server.

II. Related Work

The challenges outlined above have been collected from the research results of other teams (Table I) - and nearly all of them offered good design viewpoints during the development of our own data collecting application.

Perhaps the best overview of the possible difficulties was provided by Earl Oliver [10]. While developing a data collecting application for BlueBerry, he defined five of the most common and serious problems, namely volatile file systems on mobile devices (as file systems can be easily mounted and unmounted on nearly any device), the energy constraints, the intervention of third-party applications running in the background, the non-linear time characteristics of the devices, and malicious user activity (file manipulation, simulated manipulation, etc.).

He solved these by exploiting many trends of BlackBerry users: the general maintenance of high battery levels, retrieving manifests of active applications, and data analysis for patterns of manipulation attacks. However, even he could not define a general solution for every problem, and these problems were not the only ones encountered by other research teams. In fact, they found other challenges to be rather common among data collecting applications.

The researchers at Rice University, while developing LiveLab [11], a methodology used to measure smartphone users with a similar logging technique, encountered the problem of energy constraints, with various optimisations needed to lower the high consumption of the logging application. They also recognized the problem associated with data uploading, namely the connectivity to the server which collects the data from the devices and updates them with new information. They chose rsync for its ability to robustly upload any measurement archive which failed earlier.

A similar method of re-uploading the failed archives was used by a research team at the University of Cambridge in their Device Analyzer project [12], which sought to build a dataset that captured real-world usage of Android smartphones, again with a similar event logging based solution. They found that repeated attempts at uploading caused duplicated data on the server, which could simply be removed by the server before saving it to a database. They also solved the abovementioned problem of nonlinear time by timestamping every measurement with the device's uptime in milliseconds, recording the wall-clock time of the device when their application started, and later recording every adjustment to it by listening to the notifications caused by these adjustments. From these three elements, a simple server-side processing algorithm was able to reconstruct the exact wall-clock time of any given measurement.

Members of the Italian National Research Council [13] also confirmed these challenges (i.e., the scarcity of resources, difficulties with network monitoring and privacy) while also highlighting two more problems, caused by the participants using the devices - the much more complex control tasks in these types of research projects, and the issue of user motivation to get them carry out the tasks required to get valid data.

The collected data - even after correction - did not give satisfactory answers to the emerging questions of P2P connections with mobile devices. Therefore, we have to examine

TABLE I. COMPARISON OF DATA COLLECTING PROJECTS

Problem	*BlackBerry logger*	*Device Analyzer*	*Portolan*	*Livelab*
Energy constraints	OS callback based logging	Only 2% of the energy consumption	Computational and analyzation processes are run by the server and the collecting is not too energy consuming	The logging events are optimized, some of the data being collected directly from the system logs
Non-linear time	Dates are logged in a UTC timezone; datetime modifications recorded	Every measurement stamped with a device uptime in milliseconds; on startup, the device time is logged, like every modification on device time	Not described (there is a strict communication between client and server, probably kept in sync by this procedure)	Not described, the datetime is most likely to be among the logged data
Offline state, unsuccessful upload	-	Batched uploads only when the device is online and the charger is connected	Uploads are handled by proxy servers	Rsync protocol keeps trying until the upload is successful
Multiplicated data	-	Every device has a file on the server, multiple copies of data being detected by the server	This is solved by proxy servers	Not described most likely to be filtered by the server

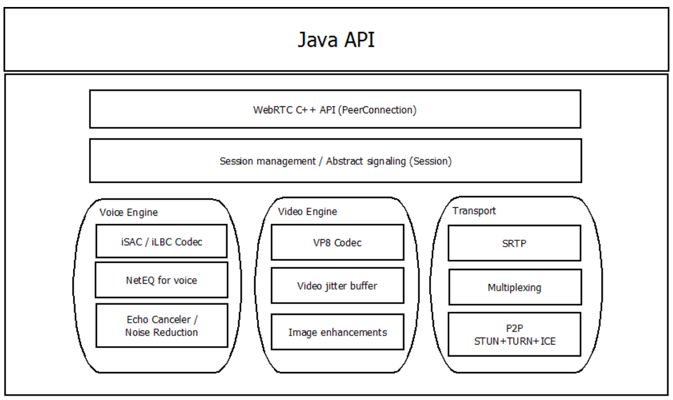


Figure 1. WebRTC Architecture

how the mobile devices behave in a real P2P environment. In most use cases the peers are computers that execute Web applications in a browser. In these environments the requirements of power and network are not as important as in mobile environments. To establish a P2P connection between two mobile devices we choose the Web Real-Time Communications (WebRTC) technology, which is an open source project that provides browsers and mobile applications with Real-Time Communications (RTC) capabilities via APIs [14]. The WebRTC exploits the concept of differentiating the media plane and the signaling plane. The signaling is managed solely on the application layer. It is so because the different applications prefer diverse signaling protocols [15]. Managing P2P connections via WebRTC, we have to handle NAT traversal and signaling. These are implemented in the Libjingle C++ library, which was integrated into the WebRTC implementation. The Libjingle library is a collection of C++ programming code and sample applications that help developers with building P2P applications. The Libjingle team created its own protocol based on Jingle and Extensible Messaging and Presence Protocol (XMPP). Figure 1 shows the architecture of WebRTC in our Android environment.

In this article, we present our experiences with crowdsourcing-based data collection along with our methods and results of data cleaning on the present dataset.

- We propose a solution for the biggest challenge of the batched data uploads, namely the time synchronisation among the different elements of the architecture, utilising a 3-way logging solution, and lightweight log synchronisation.
- We introduce heuristics to analyse incorrect NAT values, in order to decide which cases failed because of server side problems, and which cases originated from the client side.
- We also introduce a data cleaning algorithm to correct timestamp overlaps, using battery-based smartphone heuristics to detect anomalies among consecutive measurements, such as excessively rapid charging, or charging when the smartphone is in a discharging state or when no charger is connected.
- We represent a P2P architecture operating on mobile devices using WebRTC and we share the experiences gathered from establishing the connections.

III. OUR FRAMEWORK

A. Peer-to-Peer Architecture

We also needed to see how these NAT types and network information both affect a real P2P communication model. To achieve this, we have implemented a P2P module using only STUN servers and the Google Firebase Cloud Message (FCM) service. When starting the P2P measurement, the first step is to initialize a few important attributes, such as the list of STUN servers to be used and the various parameters of the communication channel. With the STUN servers, the application can determine the IP address and ports available for the outside world. After these parameters were achieved, the initiating peer has to create an offer for the Signaling Server containing the available address and port. Based on this, the signaling server (the FCM service in our case) saves the device and returns a package with the Session Description Protocol (SDP) and several packages for the Interactive Connectivity Establishment (ICE) mechanism. These packages contain all the information on the peer needed by the future partner devices for communication. The most important attributes include the *ice-pwd*, which is the device's password to the server, the *ice-ufrag* containing the username of the device, and the IP address - port combination contained by the SDP. The ICE packages also describe the protocol, which is needed to communicate with the port they represent (the two most common being TCP and UDP). For these kinds of P2P measurements, the UDP is much more suitable since the acknowledgement-based approach of the TCP is too slow and not necessary when a small degree of package loss is acceptable in the typical P2P applications (streaming, video chat, etc.). After the offer was saved on the initiating peer, it sends the SDP description and the ICE candidates to the target peer through the signaling server. The Google FCM supports multiple types of messages, from which we chose the *data message*. The main benefit of this is being a special ”silent push” message type, requiring no activity from the device owner, and being able to even wake the background service if it is in a sleeping state. The receiving peer has to know which peer is initiating the session, what is the type of the message (SDP or ICE) and naturally, the contents of the message. The connection-handling is implemented in the class called P2PService. This receives the messages, SDP packages and ICE candidates sent via push notification. The SDP is immediately saved with a response description created instantly and sent back to the initiating peer. The ICE candidates are handled by a specific algorithm. The received candidates are organized in a list, with the lower port number receiving the higher priority. The algorithm always tries to connect to the port with the highest priority, moving on through the candidates if the most recent attempt failed. The Service sends a STUN protocol message based on the chosen ICE candidate. If a response is received, the target IP address and port for the P2P session is successfully chosen. After this step, the previously initialized parameters of the communication channel are altered. Our datachannel implementation includes built-in functions for sending messages and a default listener for receiving these. The current state of the P2P session is also determined with listeners. The *localPeerConnectionObserver* in our code is an instance of the PeerConnection.Observer class, which monitors every step of the connection initializing process, the SdpObserver instance in the P2PService listens to updates concerning the SDP packages, and the DataChannel.Observer class is responsible for monitoring the state of the communication channel. The device ids are stored in a Firebase Realtime Database, this is the source from which initiating peers can choose their targets for P2P communication. Every time the application is opened, the MainActivity component sends a query to this database for the available peers. Later on, when a P2P connection is needed, a random device id is chosen from the result of this query. The list is also dynamically updated. When an instance of the Stunner application is closed, the destroying process sends a last message to the signaling server, notifying it to remove its id from the database. Similarly to the NAT measurements, the logs of the P2P connections are collected on our Parse database server. The attributes of these logs are described in Table II.

TABLE II. P2PMEASUREMENT

P2PMeasurement	
chosenID	The android ID of the device chosen by the initiating peer.
selfID	The android ID of the initiating device.
sender	A boolean value, whether or not this device was the initiator of the connection.
startConn	A timestamp for when the connection was initiated, meaning the moment of the HAVE_LOCAL_OFFER event on the initiating side and the HAVE_REMOTE_OFFER on the chosen peer
completedConn	A timestamp for when the creation of the connection finished and the devices connected.
channelOpen	A timestamp for when the state of the data channel became OPEN.
channelClosed	A timestamp for when the state of the data channel became CLOSED.
timezone	An integer, the difference between the UTC and the local time on the device.

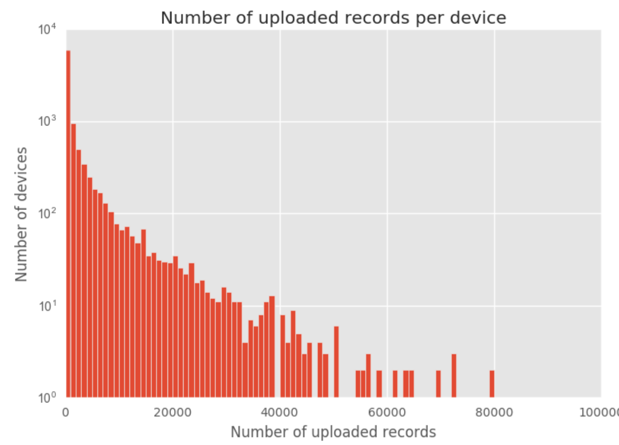


Figure 2. The plot shows the uploaded data per device

B. Statistics

Our application went live on 20th December, 2013. To promote the usage of the application we also launched a campaign, during which we provided 80 university students and users with smartphones, who agreed to download and provide data with the application for the duration of one year. On 14th February, 2018 the application had been downloaded and installed by 20,634 users on 1,184 different device types representing 1,300+ carriers and 35,000+ WiFi networks.

Our target API level was originally 19 (Android 4.4), but the application is still being downloaded and installed on more and newer devices, with Android 8.0 being currently the most popular on active installs.

We have also reached a wide variety of different types of devices, upon which the application got installed. The application also successfully reached hundreds of different mobile providers in different countries, which provided us with various, realistic NAT patterns and traces - which will be important later on, after the data cleaning phase is completed, and the analysis and usage of the data collected has commenced.

1) The collected data and the most important descriptive statistics: Based on the size of the dataset collected and our good track record since the 2013 release, it is safe to say that our application and data collecting campaign were both a success (with 70+ million records).

The chart (see Figure 2) shows the number of uploaded records per device. The majority of users did not provide any measurements, but the decline of the slope lessened, indicating that the users who provided data were more likely to stay and keep providing data.

During the summer of 2015 we had to reassign some resources to other projects, resulting in an absence of data in the given time period. However, after restarting the server, our input declined only slightly, resulting in a steady amount

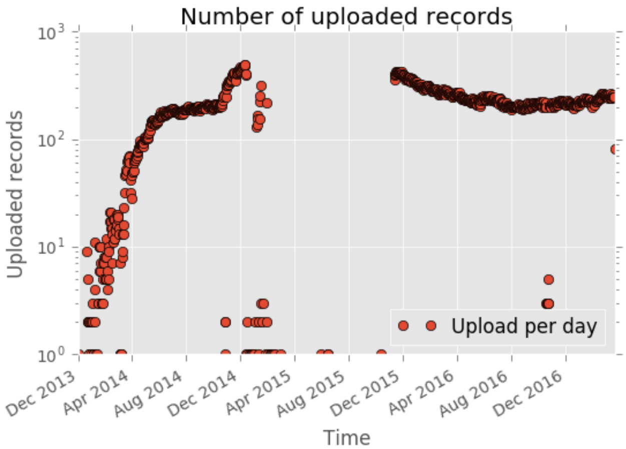


Figure 3. The plot shows the uploaded data per day during the whole measurement period.

PSEUDO CODE:
```
data['duplicated'] = = data.duplicated(subset=[all client side columns] , keep='first')
```

Figure 4. JSON sample

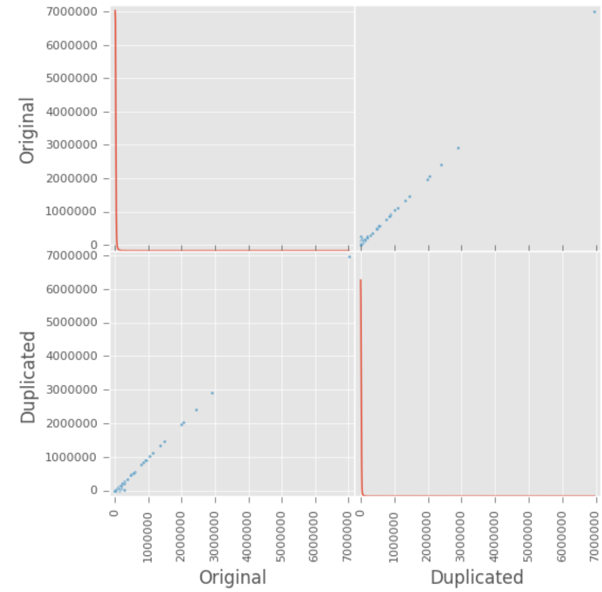


Figure 5. The plot shows the duplicated data per device vs the total number of records submitted by that device

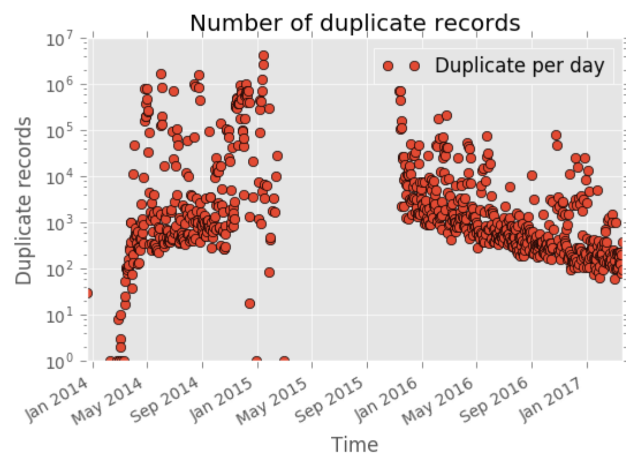


Figure 6. The plot concerning the duplicated data per day (server side)

of data arriving to this day despite a gap of a few months (Figure 3).

An interesting aspect is that although the daily uploads have been pretty steady since the hiatus, the number of active devices providing the uploads has been on a steady decline since early 2016. We hope that with our current developments, this decline can be reversed, and a new record on both the number of active devices and daily uploads can be obtained.

Following the hiatus, the first spike above shows all the collected data uploaded to the server at the same time. While the WiFi-based tests closely follow the trends of active devices and daily uploads, the Mobile Operator-based measurements have been taken at a relatively low, but steady rate. We can monitor more than 200 mobile networks and over 500 WiFi environments day after day. Here we have shown that a significant amount of data has been collected over the three-year period. The real value of the data depends on the quality of the timestamps. Now, we will describe our findings in the area of data cleaning.

IV. Issues with Collected Data

A. Data Duplication

In spite of the theoretically sound software environment where the server-side logic was implemented in JEE with transactional integrity taken into account, it turned out that a significant proportion of the dataset had been duplicated. We applied simple heuristics in order to filter out the duplicate measurement records by comparing only the client-side content and skipping the server-side timestamp and other added information. In practice, we utilized the Python Pandas framework duplicate filtering method shown in Figure 4 to remove the duplicates.

We found that out of the 70+ million rows only 30+ million rows were unique, while the remaining rows were duplicates. We investigated the possible root cause of this phenomenon. Figure 5 shows the total submitted records per device versus the duplicated records per device. It clearly shows that there is a linear relationship between the two values. This is evidence that this is a system-level symptom and not a temporal one related to server overloading. The same is true if we check the temporal dimension of the duplicated records during the given period (Figure 6).

After an in-depth investigation of the client code, we found that the default HttpClient configuration contained a very robust upload model, with a default value of 3 retries for every HyperText Transfer Protocol (HTTP) operation, if it failed with a timeout. This is a very useful method for simple data upload, but in our case, if the timeout chosen in the settings was too short, the client might have uploaded the same batch of records up to four times to the server, which would acknowledge and store all of them. In order to stop further multiple uploads, we will need to carefully look at the correct timeout and retry values and also identify the upload batches, so the server will be able to detect the retries on upload.

1) Detecting the overlap of the client-side timestamps: The actual unreliability of the client-side timestamps was a surprise for us. Figure 7 shows the difference between the Android timestamp and the date captured on the server side. A significant number of measurements have big differences between these two dates. The difference between the two

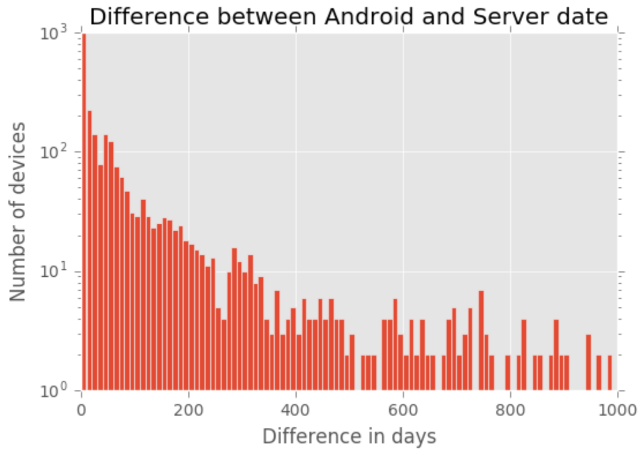


Figure 7. Difference between the Android and Server date

TABLE III. HEURISTICS FOR DETECTION

Name	*Description*	*Detection capability*
Fast change detector (ABC and SBC) (the first letter codes the ordering applied: A - android, S - server side, this coding being consistent among the different detectors)	We used the battery percentage and its sluggish behaviour to detect the fast changes. We defined the speed of change as the ratio of the two consecutive timestamps and the battery percentage difference between these two timestamps. We defined a threshold high enough to be able to recognize the measurement as an error.	For time-reset starting date estimation.
Rules based on charging and plugged state	This method focuses on the rules defined without time being included. Rules: Charging (more than 20% change) while not on charger (ACEU-SCEU) Charging (more than 20% change) while in discharging state (ACED-SCED) Big changes between consecutive elements (charging 20%, discharging 6%) (AP-SP) Charging (more than 20% change) while not on the charger and the phone is in a discharging state (ASC-SSC)	These methods could be applied in order to detect the beginning of a new measurement period (among the overlapped timestamps)

timestamps is only an indication that there could be an error in the measurements as a week or weeks may pass by after capturing and uploading the data to the server in the case of missing or inadequate network conditions.

We started to examine the nature of the Android timestamp. First, we noticed records with timestamps that were significantly earlier (e.g., 01.01.1970) or later (01.01.2023) than our other measurements. Finding invalid time periods was trivial (like 2023), but it transpired during our in-depth investigation that several phones were reset to a valid date that lay within the observation period. In order to be able to properly detect this anomaly, we elaborated several simple heuristics for detection, these being shown in Table III.

We applied the anomaly detection heuristics mentioned above in order to compare two basic sorting approaches; namely, sorting by the server-side information (e.g., serial number) and the sorting based on the mobile timestamp. We observe that for about 6-7 thousand devices the number of errors is zero. So, about 1/8 of the total devices are affected by the time overlap. Figure 8 shows the results of the fast change detector applied for the two ordering approaches (it was run on a filtered dataset, skipping the valid data). The green line (server-side sorting) indicates fewer fast change errors in most cases (it was able to eliminate this error on about 40% of the affected devices). The scatterplot below (Figure 9) also shows a clear correlation between the two sorting approaches and the number of fast change errors. The slope of the correlation line (and the points under the line) tells us that the server side sorting was able to reduce the fast change errors in most

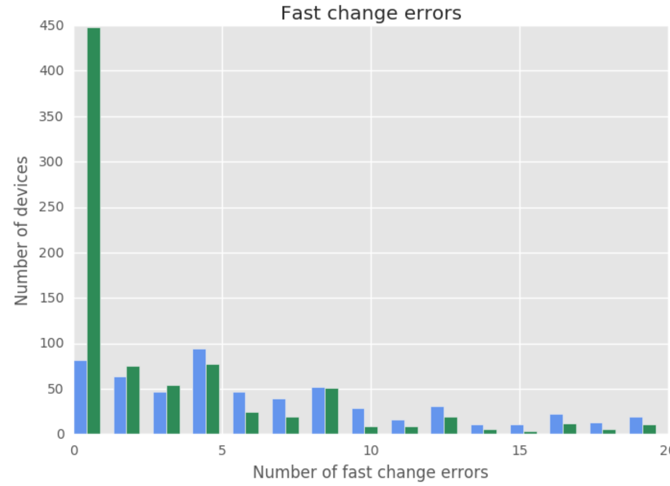


Figure 8. Fast change errors

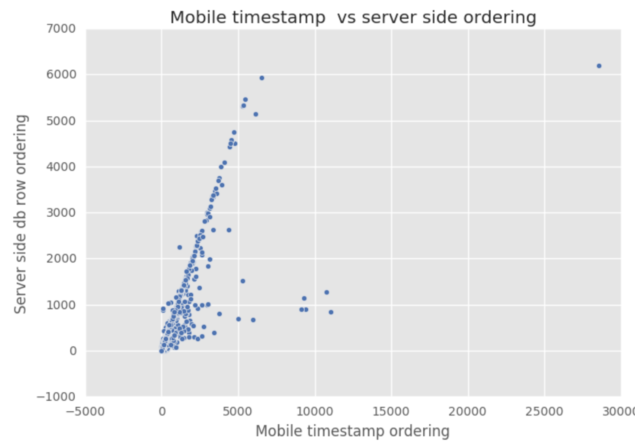


Figure 9. Mobile timestamp and server-side ordering

cases. Based on these findings, one simple approach for time overlap fixing might be the hybrid sorting approach where a given number of records are located after a fast change error had been sorted based on the server-side sorting.

The effectiveness of the simple server-side sorting is also shown in the Figure 10 concerning the correlation between different error detection heuristics and the sorting methods. It is apparent that server-side sorting can significantly decrease the error level for all error detectors (when comparing the same method with S and A sorting, most of the points are below or above the similarity line).

With the previously described heuristics, we were able to demonstrate that the server-side sort order can reduce the rows suspected of being in the wrong position to about 1/10 of the total dataset. A further decrease in the suspected errors could be achieved with a richer ruleset that incorporated different mathematical models for batteries. For our purposes, the current reliability level of the causality dimension of the dataset is quite sufficient.

B. *Sorting by client-side timestamps*

Another advanced utilisation of our collected data is interpreting the multiple user-sessions as a trace of a real network, where each user can communicate to each other in real time. In an ideal world, we may assume sorting by client-side timestamps while the client-side timers are synchronised with

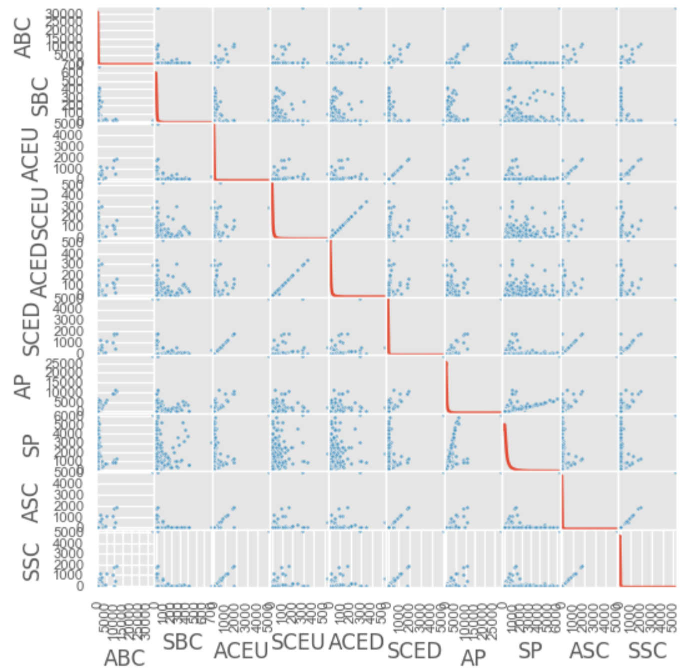


Figure 10. Error detectors

each other. However, as we mentioned above, there is a lot of uncertainty on the local timer, which is a major concern for us. In addition to this, the previously proposed methods do not fully cover this problem. This is because we need a sorting based on the local timestamps, which is in the same timeline with respect to actual time and we also need a filtering process for all the potential errors. Hence, here we will present a filtering proposal which ensures that the retained client-side timestamps are reliable.

Every examined timestamp has the same format, and it represents the number of milliseconds that has elapsed since Unix Epoch (UTC). The client-side timestamps are captured from the local timer. However, there is no further information about the local timer. All the methods below are performed for each user. In a nutshell, our solution is a three-step filtering procedure. First, we prefilter the data based on some basic rules. Second, we assume that we can trust the client-side timer and perform a sort. After, we look for clues about remaining errors. In the case of a hit the affected phase is filtered. From a bird's eye view, the method is permissive while everything seems to be normal and the method performs a systematic search for errors when an unconventional event has occurred.

At the very beginning, we perform a rule-based prefiltering. We discard all the records that have a more recent client-side timestamp than the server-side timestamp. We also discard all the records that have an older client-side timestamp than the actual appversion release date. The duplications are also prefiltered as described in Section IV-A.

Then, as we mentioned before we assume that the uncertainty about local timer is negligible. With this assumption, we can perform a sorting by client-side timestamps. Also, we record the previous position number from sorting by server-side timestamps for further examination. Now let us look at the assumption that we made. In the prefiltering phase we handled a lot of misleading measurement records, but not the whole. Therefore, our assumption is basically wrong. Hence, we need to analyse the new order and filter the remaining error.

Next, we will describe the essential part of our method. We look for a pattern that indicates the start of an error in consecutive records. If we find an error, we discard every record until any evidence is found for a certain phase. And we start the troubleshooting process repeatedly to the end. The consecutive pairs are expected to have the same incrementation for the previous and the recent sorting. Consequently, the main clue for a potential error is the difference in the incrementation of previous position number from server-side timestamp sorting. However, the examined record pair may still be in the correct order. This by itself is not a proof of an error. Therefore, further examination is needed. After, this event triggers a leap to a *questionable* state. In a questionable state, we are looking for evidence about consistency of the two consecutive records. If we do not find any error then we leave it as it is. In most cases, an error in obviously offline pairs does not bother us. This is because the order of the offline records does not have an effect on the properties of the network. Only the online ones do. To explain what we mean by a reliable pair, let us first check the uptime information of the device. If both records have one, we compare their difference against the difference between the local timestamps. A one-second deviation is tolerated. If the deviation is sufficiently small, then the pair is reliable. We continue our examination with the change in battery level. An increasing or identical battery level is the proper change when the device is charging up. Similarly, a decreasing or identical battery level is the proper change when the device is not charging up. In both cases, a one-unit deviation is tolerated in the opposite direction because of the inaccuracy of the measured battery level. A change in the charger status is not permitted, unless a proper trigger event about the change has occurred. These are very strict conditions for deciding whether a pair is reliable or not. In any other case the correctness is not guaranteed and this causes the troubleshooter to move into the *error* state, which is a substate of questionable state. We also look for many reliable pairs to end these sessions. More specifically, these phases need to be at least one hour long from the last known questionable or error pair. This is the event that triggers the end of the questionable state and the error state. Upon quitting a questionable state without an error state, the whole examined phase is left as it is. However, when quitting an error state, the first record is left, as at the end of the previous session, and the whole phase is deleted (good or bad pairs equally) up to the last questionable or error pair. From these pairs, the first is also deleted and the last will be the first record of the next session. The troubleshooting process is restarted from the end of the previously found, reliable one-hour session. Also, a one-hour difference between two consecutive records separates them. In this case, the first record ends the previous phase, and the second starts a new one.

Lastly, let us discuss the number of discarded records. As the result of our filtering method, 2.33% of the records have been erased. Thanks to our consistency check, another 0.45% of the records were temporal candidates for deletion, but they have survived.

C. NAT discovery result code corrections

The main feature of our application is the discovery of the NAT type. Users can ask the application about their NAT information and public IP address. This method is based on User Datagram Protocol (UDP) message-based communication between the device and a randomly picked STUN server. A STUN server can discover the public IP address and the type of NAT that the clients are behind.

We were faced with a problem that was caused by the prefixed STUN server list. It contains a list of 12 reliable servers that are suitable for NAT detection, this list being embedded inside the application code. It allows the device to randomly pick a STUN server. As a result, every measured NAT type in the timeline is based on a different STUN server's NAT test. Hence it makes the measured data more trustworthy. This random pick approach was well designed and worked very well initially. However, after a time four of the STUN servers went offline without any prior notice. Since then these four failed STUN servers provide the same NAT discovery result code as firewall blocked connections. As a result of this error, some uncertainty exists in the NAT discovery result code. Therefore, we propose a solution on how to correct it and make the collected data useable afterwards. Quite clearly, another solution is needed to avoid connections to a failed STUN server.

Now we need to discuss the obscure NAT discovery result code. This is the 16.76% of the total measurement records. We carried out this examination over the dataset, which had already been prefiltered and processed, the order being based on the approach defined above (see Figure 11).

Firstly we need to discuss the FIREWALL_BLOCKS result code. This code corresponds to NAT tests that have an open communication channel, but never get any response from the STUN server. In the normal case it means that the firewall blocks the connection. Unfortunately, the records also have no response from failed server, even though a part of these records may have an online NAT type. Therefore, these records are uncertain and further examination is required.

Below, we present a method for filtering the STUN server errors from the FIREWALL_BLOCKS discovery result code. This set of records contains uncertain potential online states. The server fails with a 4/12 probability, and the event of consecutive repeated fails has an exponential pattern. Consequently, we define sessions with consecutive repeated FIREWALL_BLOCKS discovery result codes and look at their distribution. If the distribution has roughly an exponential distribution, then we can interpret them as online and we can define their network properties. Otherwise, those that do not have an exponential fit will remain FIREWALL_BLOCKS. This means that in this way we cannot prove the opposite (the firewall blocks the connection). In general, we are looking for a session that begins and ends with the same network property and there are only uncertain online states between them. These sessions may be interpretable based on the begin-end enclosures. More specifically, the sessions must

- begin and end with the same NAT discovery result code
- begin and end with the same Service Set Identifier (SSID) in the case of a WiFi connection
- begin and end with the same mobile operator in the case of a mobile data connection
- contain only uncertain online states
- contain a time gap between two records only in a range of 0 to 15 minutes based on the fact that the maximum time gap between two regular online records is almost 10 minutes. However, it is not very accurate because of the Android support scheduler with its inexact trigger time requirements.
- not be interrupted by trigger events that correspond to any potential change in the network properties.

We show the above-defined candidate sessions in Figure 13 and Figure 14. Let us first take a look at how many uncertain discovery result codes are enclosed by these sessions in Figure 13. It is clear that the first four points seem to fit an exponential curve. Consequently, it is still open to interpretation and the rest of the points remain undefined. Next, Figure 14 shows length of the above-defined sessions. There are some peaks every 10 minutes or so. These peaks correspond to the BATTERY_SCHEDULED trigger event, which is scheduled every 10 minutes and this is the most common trigger event. For example, if there is exactly one uncertain FIREWALL_BLOCKS value in the appropriate session and every taking of a measurement is triggered by this schedule event, then its length of time is around 20 minutes. Based on this example, an above-defined session that contains three unknown records lasts for 50 minutes. Accordingly, we examined the points from the first phase up to 50 minutes. Our examination revealed that it also had an exponential pattern. In contrast to the distribution in Figure 13, this distribution appears more complex, but it is still acceptable. Next, we associate the two findings. More specifically, the intersection of the two sets is an above-defined session that contains fewer than five uncertain elements and it lasts no longer than 50 minutes. Based on this rule, we can correct the network properties of 6.7% of the measurement records.

Next, we should mention some other minor errors associated with data collection. In very few cases there was no network connection, but it still has some errors in the discovery result code (mainly code 0). Here, we simply correct all of them to the no-connection state (-2).

Now let us have a look at the final results of the NAT data correction in Figure 12. The records with FIREWALL_BLOCKS code have dropped to 10%, and the records with an online state have increased.

V. Lessons learnt

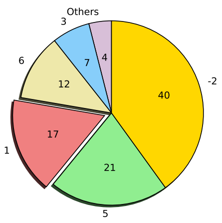
A. NAT measurements

Based on our findings, some of the challenges encountered proved to be quite trivial, and required only some small modifications to the algorithm, while others still have to be tested with our proposed solutions.

On the client side, we have found several elements where the default approach of Android development proved insufficient, and special consideration was needed for proper data collection. We found that the timeout value of the Android application should be increased in proportion to the connectivity quality with the data collector server, while the number of retries should be reconsidered and perhaps revised with upload batches accompanied by identifiers to make duplicate detection easier.

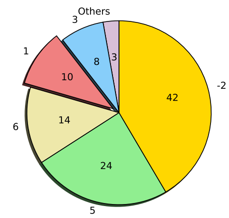
The detection of the NAT anomalies was made significantly easier through the NetworkInfo and WifiInfo objects of the Android system. When collecting network data, we found it highly advisable to include as many attributes from these rich objects as possible - such as SSID, whether the phone is in the roaming mode and whether the network is connection metered -, since any of these could explain possible anomalies in the dataset. For example, the phone might be connected to a WiFi network, but the router is not necessarily connected to the Internet; or, if it is located at a public establishment, it may redirect the requests to the establishment's login site instead of the original destination - all of which are serious problems, and they could go unnoticed without detailed information about the network.

Regarding the NAT problem, it is also advisable to reconsider storing the list of external servers in a constant array



NAT Discovery Result Code			
-2	NO_CONNECTION	11951611	40.03%
-1	ERROR	29	< 0.01%
0	OPEN_ACCESS	706662	2.37%
1	FIREWALL_BLOCKS	5002770	16.76%
2	SYMMETRIC_FIREWALL	198442	0.66%
3	FULL_CONE	1993739	6.68%
4	RESTRICTED_CONE	280969	0.94%
5	PORT_RESTRICTED_CONE	6240586	20.90%
6	SYMMETRIC_CONE	3482591	11.66%

Figure 11. Discovery Result Code



NAT Discovery Result Code after correction			
-2	NO_CONNECTION	12401805	41.53%
-1	ERROR	29	< 0.01%
0	OPEN_ACCESS	284206	0.95%
1	FIREWALL_BLOCKS	2999985	10.05%
2	SYMMETRIC_FIREWALL	230693	0.77%
3	FULL_CONE	2295290	7.69%
4	RESTRICTED_CONE	318136	1.07%
5	PORT_RESTRICTED_CONE	7272018	24.35%
6	SYMMETRIC_CONE	4055237	13.58%

Figure 12. NAT Discovery Result Code after correction

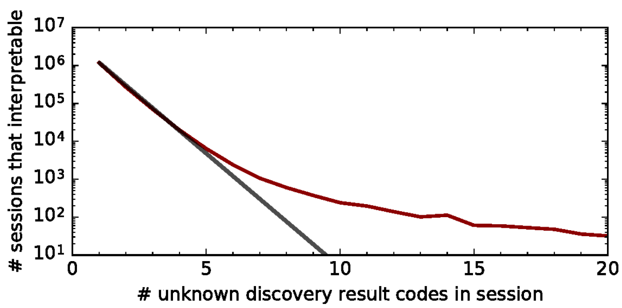


Figure 13. Discovery result code enclosed by sessions

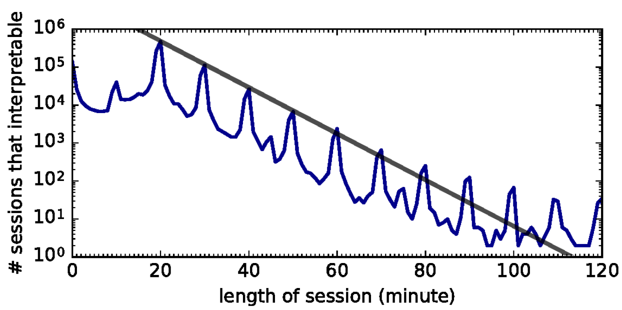


Figure 14. Length of candidate sessions

(a practice which is very common based on our experiences), because if one of those servers goes offline, it might generate huge amounts of incorrect data. A proxy which stores the server list, keeps it updated by using regular checkups, and forwards the list to the phones on request, would be a better solution here.

Also, while the deletion of previous data is a good practice to stop the application from taking up too much storage space, the 24-hour limit might be too short, since important events could get lost in that time period. The time limit for storage before deletion should be featured among the settings. Even after a delete, it is necessary to leave some trace of the deleted data - at least a log -, so the anomalies in the later, successful uploads could be interpreted.

The timestamp desynchronisation between the server and the client remains perhaps the most challenging problem, with the battery-based sortings providing some improvements in the dataset. One solution might be a lightweight log timestamping. In this case only a hash of the log would be sent to the server frequently (in order to minimise the mobile traffic and save the battery), where a reliable timestamp would be attached on the server side to this hash and saved in a permanent storage. In this way, we may define reliable milestones which are independent of the mobile side timestamps. On the mobile side, it is important to preserve the total order of the events. This could be achieved by using a simple increasing indexing procedure in the SQLite database.

We mentioned earlier that even NAT types may be misleading, despite the quality of the connection. Once again, some of these incorrect values could be corrected by simply checking the actual state of connectivity during the upload. The NAT type in the remaining records is mostly corrected by a pattern recognition method. Hopefully, this problem may never occur again after the proposed changes have been made to handle a dynamic STUN server list.

B. *Peer-to-Peer measurements*

After the first implementation of the P2P connection we wanted to know about the time of establishing a connection, making a disconnection and detecting the peer's disappearance. In addition, we wanted to know what happens to the peers if they make a reconnection or if they connect to another access point.

In order to discover the capabilities of WebRTC and understand how the peers work in practice, we have to keep the whole signaling mechanism in hand. These expectations address several challenges.

Firstly, we want the measurements not to be influenced by users, so we cannot use the normal push notifications. Firebase Cloud Messaging (FCM) provides a solution for sending notifications without visible alerts, this is the so-called silent push notification. With the help of FCM, we can send data to the application that runs on a device; however, the process does not need user interaction.

Using the silent push notification, we have recognized other issues. This type of notification can wake up a device if the application runs in the background. If we kill the application that runs in the background, the silent push cannot address

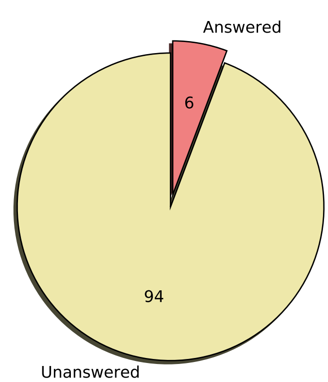
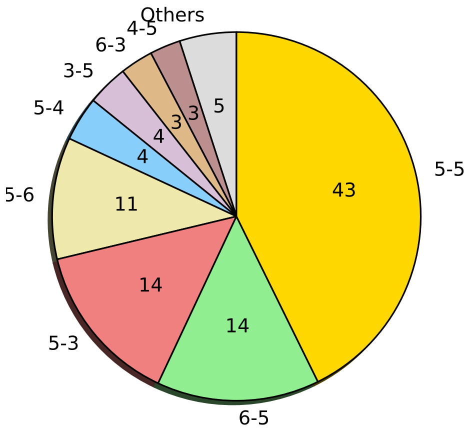


Figure 15. Ratio of answered and unanswered offers



Caller's NAT type	*Callee's NAT type*	*Ratio of P2P connections*
Open Access (0)	Full Cone (3)	2.5%
Symmetric Firewall (2)	Full Cone (3)	1.8%
Full Cone (3)	Port Restricted Cone (5)	3.6%
Restricted Cone (4)	Port Restricted Cone (5)	2.7%
Port Restricted Cone (5)	Open Access (0)	0.7%
Port Restricted Cone (5)	Full Cone (3)	14.3%
Port Restricted Cone (5)	Restricted Cone (4)	3.9%
Port Restricted Cone (5)	Port Restricted Cone (5)	42.8%
Port Restricted Cone (5)	Symmetric Cone (6)	10.7%
Symmetric Cone (6)	Full Cone (3)	2.9%
Symmetric Cone (6)	Port Restricted Cone (5)	14.3%

Figure 16. Number of P2P connections with different NAT types

the device. When the application starts, the mobile device subscribes to the Firebase Realtime Database. Hence, we store the set of available devices in this database with a hashed identifier. If the user quits from the application, the device unsubscribes from the database. Since killing an application is a different event compared to the sophisticated quit method, the onDestroy() method will never be called. Hence, we had to catch the small time gap that occurs right before killing the application.

During the tests we have also discovered that the resources for an application that runs in the background are only allocated for a given time. Unfortunately, we cannot catch the exact timestamp of deallocation, so the devices that take the resources away from Stunner are shown as still available, but the notification cannot wake up them. We also have to take into account this anomaly.

Under the examination of the collected logs, we have also realized that in many cases the SDP information are not satisfying to establish P2P connection. In some cases, only IPv6 information are retrieved from the STUN server and the necessary IPv4 information are missing. Sometimes the public IP address and available ports are missing and only the private IP address is given. Due to lack of necessary information, the P2P connection cannot be established between two peers.

Knowing these issues, we have cleaned the collected P2P logs first, then we started to analyze the remaining data. The first version of Stunner - that contains the P2P module - was published on 25th April, 2018. Since then we have released 3 different updates that improved the performance of the application a lot. Since the new version of Stunner was released, we have collected more than 200,000 logs.

The collected data showed that more than 90% of the offers were unsuccessful because the silent push notification could not awake the slept application. As it can be seen in Figure 15, the number of sent offer dominates the number of sent answers. The ratio was calculated from the Firebase server logs.

After the data cleaning, we still have a huge number of useful data. From the remaining data, we have seen that the peers behind different NAT types show an interesting pattern to the P2P connections. Figure 16 shows ratios of the successful P2P connections. An interesting fact is that most of the successful P2P measurements were done by devices behind Port restricted NATs. Surprisingly, we have also experienced that in many cases, the mobile devices with Symmetric Cone NAT could also connect to other devices.

Another interesting point is how much time the P2P connection takes to establish. We determined this by logging four different timestamps. The first one is the time when the peer started to gather its information from the STUN server, and the second one is the time when both the peers shared their information with each other. The third one is the beginning of the time when the datachannel is created and the fourth one is the time when the connection is totally closed. The disconnection is a very fast process, but establishing a connection takes 6.745 seconds on average. It takes this much time, because the device that begins the connection has to turn to the STUN server for getting the information about the IP addresses and available ports, get a list of available peers, choose one from the list, send the data via signaling server and create the datachannel between the two WebRTC clients.

VI. Conclusions

As the reader can no doubt see, our approach worked well in the above-mentioned areas of data cleaning. Since the application was launched in 2013, it has been downloaded by more than 20,000 users from over 1,300 different carriers and 35,000 different WiFi areas, to hundreds of different device types, which is providing enormous amounts of valuable data for the analysis of NAT traces, patterns, and later on, for the simulation of the above attributes.

Compared to other crowdsourcing projects, our crowdsourcing approach was a hybrid methodology, where we provided a certain number of users with smartphones, and released the app to the Play Store for wider availability, and took more data measurements from different parts of the world. We did not reach the volume of OpenSignal or Bredbandskollen with their 100-200+ million datasets, but this hybrid solution still provided us with a much bigger amount of valuable data than a closely monitored environment like FlierMeet or SignalLab that had roughly 40 devices, a shorter collection time period, and operated in a restricted environment like a university campus or a development environment.

Lastly, we have implemented the WebRTC project in our application, and have started gathering useful information about the various metrics of P2P connections. With this data, we will be able to create an even better smartphone simulation network, where the events and metrics will be able to reflect the actual, real-life behaviour of the test algorithms. Our next major step will be the in-depth testing of P2P algorithms, finding better solutions for the keep-alive issues of the application and the determination of the speed and stability of smaller applications running in a simulated environment.

ACKNOWLEDGMENT

This research was supported by the Hungarian Government and the European Regional Development Fund under the grant number GINOP-2.3.2-15-2016-00037 ("Internet of Living Things").

REFERENCES

[1] Z. Szabó, V. Bilicki, Á. Berta, and Z. R. Jánki, "Smartphone-based Data Collection with Stunner Using Crowdsourcing: Lessons Learnt while Cleaning the Data," in ICCGI 2017, The Twelfth International Multi-Conference on Computing in the Global Information Technology, July 2017, pp. 28–35.

[2] G. Chatzimilioudis, A. Konstantinidis, C. Laoudias, and D. Zeinalipour-Yazti, "Crowdsourcing with smartphones," IEEE Internet Computing, vol. 16, no. 5, 2012, pp. 36–44.

[3] I. Burguera, U. Zurutuza, and S. Nadjm-Tehrani, "Crowdroid: behavior-based malware detection system for android," In Proceedings of the 1st ACM workshop on Security and privacy in smartphones and mobile devices, October 2011, pp. 15–26.

[4] B. Guo and et al., "FlierMeet: a mobile crowdsensing system for cross-space public information reposting, tagging, and sharing," IEEE Transactions on Mobile Computing, vol. 14, no. 10, 2015, pp. 2020–2033.

[5] M. Stevens and E. D'Hondt, "Crowdsourcing of Pollution Data using Smartphones," in Workshop on Ubiquitous Crowdsourcing, held at Ubicomp '10, September 2010, pp. 1–4.

[6] A. Leemann, M. J. Kolly, R. Purves, D. Britain, and E. Glaser, "Crowdsourcing language change with smartphone applications," PloS one, vol. 11, no. 1, 2016, pp. 1–25, e0143060.

[7] T. Linder, P. Persson, A. Forsberg, J. Danielsson, and N. Carlsson, "On using crowd-sourced network measurements for performance prediction," in In Wireless On-demand Network Systems and Services (WONS), 2016 12th Annual Conference on. IEEE, January 2016, pp. 1–8.

[8] A. Overeem and et al., "Crowdsourcing urban air temperatures from smartphone battery temperatures," Geophysical Research Letters, vol. 40, no. 15, 2013, pp. 4081–4085.

[9] Á. Berta, V. Bilicki, and M. Jelasity, "Defining and understanding smartphone churn over the internet: a measurement study. In Peer-to-Peer Computing (P2P)," in 14-th IEEE International Conference on. IEEE, September 2014, pp. 1–5.

[10] E. Oliver, "The challenges in large-scale smartphone user studies," In Proceedings of the 2nd ACM International Workshop on Hot Topics in Planet-scale Measurement, June 2010, p. 5.

[11] C. Shepard, A. Rahmati, C. Tossell, L. Zhong, and P. Kortum, "LiveLab: measuring wireless networks and smartphone users in the field," ACM SIGMETRICS Performance Evaluation Review, vol. 38, no. 3, 2011, pp. 15–20.

[12] D. T. Wagner, A. Rice, and A. R. Beresford, "Device Analyzer: Large-scale mobile data collection," ACM SIGMETRICS Performance Evaluation Review, vol. 41, no. 4, 2014, pp. 53–56.

[13] A. Faggiani, E. Gregori, L. Lenzini, V. Luconi, and A. Vecchio, "Smartphone-based crowdsourcing for network monitoring: Opportunities, challenges, and a case study," IEEE Communications Magazine, vol. 52, no. 1, 2014, pp. 106–113.

[14] A. B. Johnston and D. C. Burnett, WebRTC: APIs and RTCWEB Protocols of the HTML5 Real-Time Web. Digital Codex LLC, 2012.

[15] M. H. Rahaman, "A Survey on Real-Time Communication for Web," Scientific Research Journal (SCIRJ), vol. 3, no. 7, 2015, pp. 39–45.

Measurement and Generation of Diversity and Meaningfulness in Model Driven Engineering

Adel Ferdjoukh
LS2N & University of Nantes, France
adel.ferdjoukh@univ-nantes.fr

Florian Galinier
IRIT, University Paul Sabatier, Toulouse. France
florian.galinier@irit.fr

Eric Bourreau, Annie Chateau and Clémentine Nebut
Lirmm, CNRS & University of Montpellier, France
lastname@lirmm.fr

***Abstract*—Owning sets of models is crucial in many fields, so as to validate concepts or to test algorithms that handle models, model transformations. Since such models are not always available, generators can be used to automatically generate sets of models. Unfortunately, the generated models are very close to each others in term of graph structure, and element naming is poorly diverse. Usually, they badly cover the solutions' space. In this paper, we propose a complete approach to generate meaningful and diverse models. We use probability simulation to tackle the issue of diversity inside one model. Probability distributions are gathered according to domain quality metrics, and using statistical analysis of real data. We propose novel measures to estimate differences between two models and we provide solutions to handle a whole set of models and perform several operations on these models: comparing them, selecting the most diverse and representative ones and graphically observe the diversity. Implementations that are related to difference measurement are gathered in a tool named COMODI. We applied these model comparison measures in order to improve diversity in Model Driven Engineering using genetic algorithms.**

***Index Terms*—Model Driven Engineering; Intra-model Diversity: Inter-model Diversity; Meaningfulness of Models; Counting Model Differences.**

I. Introduction & Motivations

The present paper is an extended version of the paper that the authors published in ICSEA'2017 conference [1].

The increasing use of programs handling models, such as model transformations, more and more increases the need for model benchmarks. Elements of benchmarks are models expected to respect three quality criteria at the same time. First, they must be as representative as possible of their domain-specific modelling language. Then, they have to be diverse in order to chase the potentially rare but annoying cases where programs show a bad behaviour. Finally, they must be meaningful and close to hand-made data in some aspects. The difficulty of finding real test data that fulfil all these requirements, and in sufficient quantity to ensure statistical significances, leads to consider the automated generation of sets of models. Many approaches and tools can be used in this purpose: *Ferdjoukh et al.* [2], *Sen et al.* [3], *Cabot et al.* [4], *Gogolla et al.* [5].

We distinguish two different concepts of diversity. *Intra-model diversity* ensures that the elements contained in a single model are sufficiently diverse. For example, when generating a skeleton of a Java project, we verify that there are many packages in the project. The packages should contain diverse numbers of classes. Classes should contain fields and methods of diverse types, etc. *Inter-model diversity* forces two models taken as a pair to be as different as possible. With a possible extension to a whole set of models. The combination of these two different kinds of diversity guarantees both the well coverage of the meta-model's solution space and the meaningfulness of the generated models.

Intra-model diversity is linked to the concept of meaningfulness for models. Thus in the previous example, let us consider that the classes of a Java project are all concentrated in one package. We estimate that this project is not well-formed or meaningful. In practice, this kind of project does not respect *Object Oriented Programming* quality criteria, because its reusability is poor. In this paper, we propose to use probability distributions related to domain-specific quality metrics to increase the intra-model diversity.

One of the main issues when attempting to produce different models, is to state in what extent, and according to which criteria, the models are actually "different". The most natural way to formalize this notion, is to define, and use metrics comparing models, and measuring their differences.

Determining model differences is an important task in Model Driven Engineering. It is used for instance in repositories for model versioning. Identifying differences between models is also crucial for software evolution and maintainability. However, comparing models is a difficult task since it relies on model matching. This latter can be reduced to graph isomorphism that is an NP-intermediate problem (*No polynomial algorithm is known for Graph isomorphism problem, but it has not been proven NP-complete. The more general sub-graph isomorphism problem is NP-complete.*) [6]. Kolovos et al. [6] draw up an overview of this issue and give some well-known algorithms and tools for model comparison. Most of these approaches compare only two models between them and find their common elements. This is insufficient for the problem of diversity improving because differences have to be measured and a whole set of models has to be considered.

In this work, we propose a distance-based approach to measure model differences and we provide solutions to handle sets of models in order to compare them and to extract the most representative models. A human readable-graphical viewing is also given to estimate the diversity of a set of models.

We consider models, which are conform to meta-models, according to the Ecore/EMF formalization [7]. Model generation is performed using $\mathcal{G}$RIMM [2], [8], [9], [1] and it is based on Constraint Programming [10]. Basically, $\mathcal{G}$RIMM reads a meta-model and translates all elements of the meta-model into a Constraint Satisfaction Problem (CSP). A CSP solver is then used to solve the obtained constraint network, leading to one or more models, which are conform to this meta-model, and meeting a given set of additional parameters describing the characteristic of desired models. Constraint Programming provides a deterministic behavior for the generation, it is then difficult to encode diversity directly in the heart of the tool. Other model generation tools can be coupled with our approach. For example, during our experiment we also used models that have been generated using PRAMANA tool (Sen et al. [3]). We investigate in this work the use of genetic algorithms, so as to diversify an obtained set of models (the preliminary elements for the approach can be found in [11]).

In summary, our contributions are: (1) an approach for increasing intra-model diversity based on probability simulation (2) two case studies to show the efficiency of the previous method (3) novel metrics measuring model differences using distances coming from different fields (data mining, code correction algorithms and graph theory) and adapted to MDE (Model Driven Engineering) (4) solutions to handle a whole set of models in order to compare them, to extract the most representative models inside it and to give a graphical viewing for the concept of diversity in MDE (5) A tool implementing these two previous contributions (6) an application of these distance metrics in improving inter-model diversity in MDE using a genetic algorithm.

The rest of the paper is structured as follows. Section II relates about previous literature. Section III presents $\mathcal{G}$RIMM, our methodology and tool for automated generation of models. Section IV describes how we face the issue of intra-model diversity in MDE. Two case studies show the efficiency of the previous approach (Section V). Section VI details the considered model comparison metrics. Section VII details the solutions for handling a set of models (comparison, selection of representative model and graphical viewing). The tool implementing these contributions is described in Section VIII. An application of our method to the problem of inter-model diversity in MDE is shown in Section IX. Section X concludes the paper.

II. RELATED WORK

This section draws a state of the art of the topics related to the work we presented in our paper. Related work is divided into three parts. Each part relies on one important contribution of this paper.

A. Meaningful models

We give here a short list of approaches for model generation. Many underlying techniques have been used. *Cabot et al.* [12] translate a meta-model and its OCL constraints into constraint programming. In [13], *Mougenot et al.* use random tree generation to generate the tree structure of a model. *Wu et al.* [14] translate a meta-model into SMT (Satisfiability Modulo Theory) in order to automatically generate conform models. In [15], *Ehrig et al.* transform a meta-model into a graph grammar, which is used to produce instances. The advantages and drawbacks of our original approach relatively to the other generating methods have been discussed in [8].

Nevertheless, only two approaches have treated the problem of meaningfulness of generated models. In [13], authors use a uniform distribution during the generation process and add weights in order to influence the frequency of appearance of different elements. In [14], authors describe two techniques to obtain relevant instances. The first one is the use of partition-based criteria, which must be provided by the users. The second one is the encoding of common graph-based properties. For example, they want to generate acyclic graphs, *i.e.,* models.

B. Model comparison

The challenging problem of model comparison was widely studied, many techniques and algorithms were proposed for it. Two literature studies are proposed in [6] and [16]. Among all the techniques, we describe here the techniques that are close to the model distance algorithms we propose, in both comparison and objective.

Falleri et al. [17] propose a meta-model matching approach based on *similarity flooding* algorithm [18]. The goal of this approach is to detect mappings between very close meta-models to turn compatible models, which are conform to these meta-models. The comparison algorithm detects two close meta-models. A transformation is then generated to make the models of the first meta-model conform to the second one. However, in such kind of work, the similarity between models cannot be detected without using the names of elements: lexical similarities are propagated through the structure to detect matchings. Our approach is more structural.

Voigt and Heinze present in [19] a meta-model matching approach. The objective is very close to the previous approach. However, the authors propose a comparison algorithm that is based on graph edit distance. They claim that it is a way to compare the structure of the models and not only their semantics as most of techniques do.

C. Diversity of models

Cadavid et al. [20] present a technique for searching the boundaries of the modeling space, using a meta-heuristic

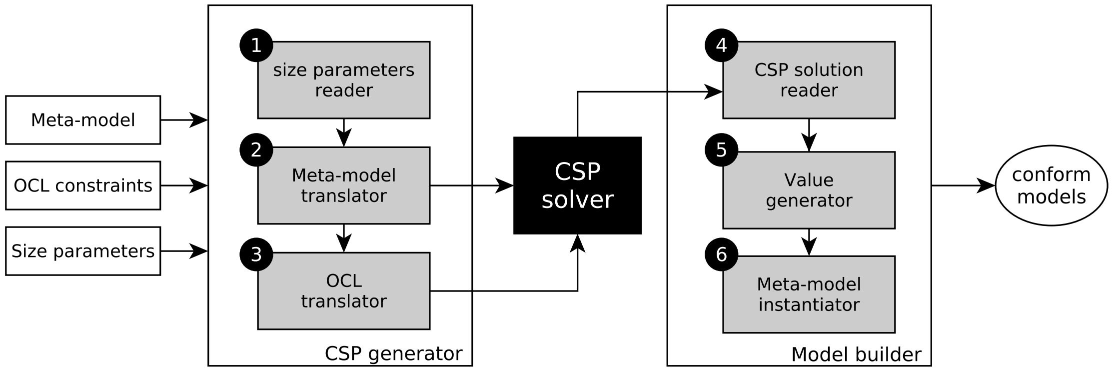


Fig. 1. Steps for model generation using GRIMM tool.

method, *simulated annealing*. They try to maximize coverage of the domain structure by generated models, while maintaining diversity (dissimilarity of models). In this work, the dissimilarity is based on the over-coverage of modeling space, counting the number of fragments of models, which are covered more than once by the generated models in the set. In our work, the objective is not to search the boundaries of the search space but to select representative and diverse elements in the whole search space.

More recently, Batot et al. [21] proposed a generic framework based on a multi-objective genetic algorithm (NSGA-II) to select models sets. The objectives are given in terms of coverage and minimality of the set. The framework can be specialized adding coverage criterion, or modifying the minimality criterion. This work of Batot et al. confirms the efficiency of genetic algorithms for model generation purposes. Our work is in the same vein but focuses on diversity.

Sen et al. [22] propose a technique on meta-model fragments to ensure that the space of solutions is well-covered. The diversity is obtained by diversifying the possible cardinalities for references. For example, if a reference has cardinality 1..*n*, 3 different configurations are considered: 1, 2 and *n*. To generate a model, the different configurations for all references are mixed.

Hao Wu [14] proposes an approach based on *SMT (Satisfiability Modulo Theory)* to generate diverse sets of models. It relies on two techniques for coverage oriented meta-model instance generation. The first one realizes the coverage criteria defined for UML class diagrams, while the second generates instances satisfying graph-based criteria.

Previous approaches guarantee the diversity relying only on the generation process. No post-process checking is performed on generated model sets to eliminate possible redundancies or to provide a human-readable graphical view of the set.

III. BACKGROUND: GRIMM TOOL

GRIMM (GeneRating Instances of Meta-Models) is a model generation approach and tool. Its goal is the automated generation of instances that conform to ecore meta-models. It is based on Constraint Programming (CSP) [10]. Schema on Figure 1 shows the steps for model generation using GRIMM tool.

A. Input

Meta-model is written using the *Ecore* formalization. It mainly contains classes, attributes and references (links between two classes). A class can have one or more subclasses and also one or more super-classes. A reference between two classes can be bi-directional or unidirectional only. Some references are compositions. One class of the meta-model plays the leading role of root class. It has the particularity of being linked to all other classes by composition relations (directly of transitively).

OCL constraints are additional constraints that come with a meta-model. They help to disambiguate some elements of the meta-model. OCL is a first order logic language. In our work we are considering only invariants and not all the types of OCL constraints. The area of OCL our tool can handle is large but not complete. We can process simple OCL constraints on one or more attributes, many operations on collections and some typing operations. It is also possible using CSP to take into account some complex and nested OCL constraints.

Size Parameters fix the size of desired generated models. These parameters are chosen by the user. He fixes the number of instances for each class and unbounded references and also can give the domain of values for an attribute.

B. Processing

Size parameters reader This module of our tool reads a file containing all the size parameters given by a user and verifies their well-formedness. It also does some inference when information is missing (for attributes values and for unbounded references).

Meta-model translator The goal of this step is to create a Constraint Satisfaction Problem (CSP) to solve the model generation problem. This module uses the previous size parameters and translates each element of the meta-model (class, attribute, link, inheritance) into its equivalent in CSP. Much more details about the translation process can be found in [2], [8].

OCL translator This step is devoted to the translation of OCL constraints into CSP. The CSP obtained in the previous step is completed to integrate OCL. Many constructions of the OCL language are taken into account. Simple and complex constraints with one or more attributes are handled. Many widespread operations on collections or on types are also considered. In addition, processing of some nested constraints is possible using Constraint Programming. Translation of OCL constraints is detailed in [8].

CSP solver It is a program that solves CSP. Its goal is to quickly find a solution satisfying all the constraints at the same time, yielding difficulties when trying to generate diverse and different solutions. Indeed, a CSP solver is designed to quickly find a solution without considering its intrinsic quality or a global vision of all the possible solutions. In other words, although a solution is correct for a CSP solver, it is not necessarily a meaningful model.

CSP solution reader Once a solution is found, $\mathcal{G}$RIMM reads this solution and extracts the values for the model's elements: inheritance, links and attributes.

Value generator The values of some elements are missing after the previous step. Because CSP handles only integer variables. For other data types: enumerations characters, strings and floats, post-processing is needed. This steps is then crucial to get a complete model.

Meta-model instantiator This final step gathers all information of previous step and builds a valid and conform model.

The result of this process is the generation of instances for meta-models. $\mathcal{G}$RIMM tool is able to produce two different outputs:

- *xmi* model files, that can be used for testing model transformations.
- *dot* is a format for creating graphs. Our tool uses its own syntax, so models can be graphically visualized in *pdf* or *png*.

The following sections describe contributions for improving the meaningfulness and the diversity of generated models.

IV. Intra-diversity of models

One remaining issue is related to the meaningfulness of generated models. Indeed, we have to produce models that are as realistic as possible, regarding to the data they are supposed to simulate. We propose to use *intra-model diversity* to achieve this goal.

The declarative approach (CSP) is intrinsically deterministic, since the solver follows a deterministic algorithm to produce a unique solution. The CSP solver can easily produce thousands of solutions, but they are often far from the reality. In other words, it is impossible using a combinatorial approach, such as CSP, to express the meaningfulness. It is more a probabilistic concept.

Generation process is enriched with various parameters. We exploit the flexibility of CSP and the fact that some elements of the real models follow usual probability distributions. These distributions are simulated and, *a priori*, injected to the CSP, in order to produce generated models closer to real ones.

For better results, probability distributions are not arbitrary. They are domain-specific. Indeed, we observe, that many quality metrics are encoded in the meta-models. For example, in Java meta-model, we find an unbounded reference (0..∗) between *class* and *method*. It determines the number of methods belonging to a given class. This number is fixed by the CSP solver according to user's choice. In real models, the number of methods per class is diverse, not constant, and there exist related quality metric [23] and code smells [24]. Our idea is to exploit the metrics, that are encoded in the meta-model, to improve the meaningfulness of generated models.

Schema on Figure 3 shows the steps for generation of meaningful models, by exploiting domain specific metrics, and related probability distributions.

A. Input

Probability distributions on links formally define the diversity, while linking class instances inside the generated models. For each reference, identified as related to a quality metric, the user defines a probability law (normal, exponential, etc). This law is then used to fix the number of target elements for each class instance. Figure 2 illustrates the diversity, while linking the elements of a model. In the left side, we can see a theoretical model without any diversity. Each *circle* is always linked to 2 *squares*. The right side shows a model, in which each *circle* is linked to a different number of *squares*.

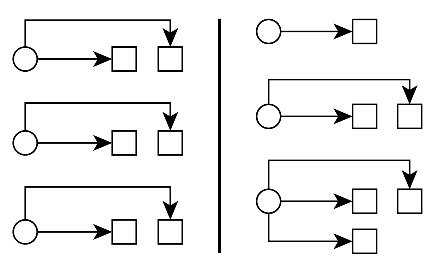


Fig. 2. Illustrating the diversity while linking elements in a model.

Probability distributions on attributes concern the most important attributes in a meta-model. For example, in the Java meta-model, the visibility of fields and methods is important. Indeed, object-oriented programming promotes encapsulation. This means that most of fields are *private* and most of methods are *public*. The probability laws determine here the value of attribute *visibility* for each created field and method. A probability is defined for each possible value, and simulation will choose adequate values and assign them to class instances.

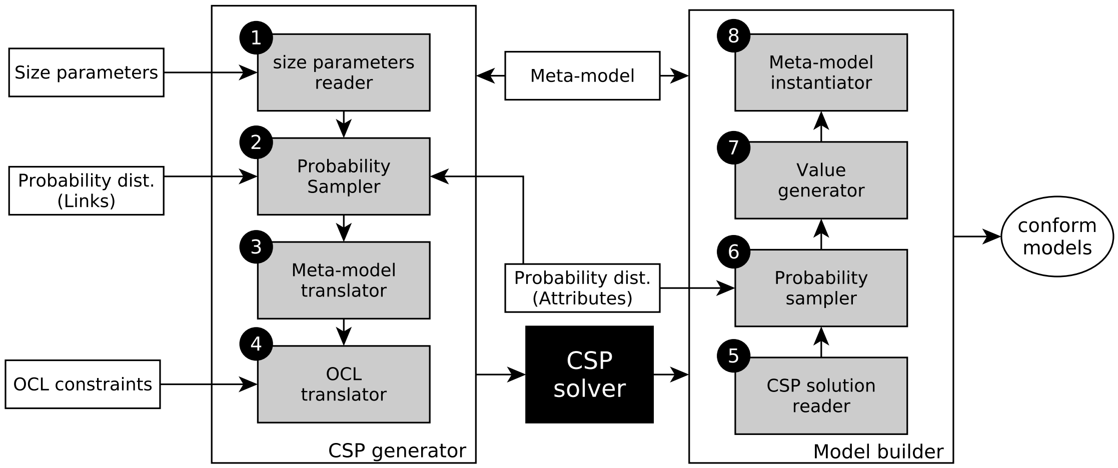


Fig. 3. Steps for meaningful model generation using $\mathcal{G}$RIMM tool.

The parameters for probability distributions are very important for the generation process. Our approach allows the user to choose them arbitrary. Nevertheless, we think that it is preferable to infer them by conducting an empirical study over existing data, or to use domain quality metrics.

B. *Probability sampler*

Generating samples of well-known probability distributions is a way to add randomness to the deterministic CSP solving process. The idea is to get models that have more diversity in their elements' degrees and their attributes' values in order to cover a lot of possible values. For example, when generating UML models, we want to generate a package, which has 5 classes, another one with 7 classes and so on.

Figure 4 shows the basic operation with which we can sample all usual probability distributions whatever they are continuous or discrete. Thus, to generate a sample of a random variable X we need its cumulative function $F(X)$ and a sample of uniform values u. Result values x are obtained by an inversion of F: $u = F(x) \Rightarrow x = F^{-1}(u)$.

Previous method is then adapted to each probability distribution we want to sample.

Discrete distribution on a finite set For all discrete distribution, are given the probabilities of a finite set of values. The cumulative function is then deduced from the accumulation of probabilities and a sample can be easily generated.

Inverse cumulative function method This method is used for continuous distribution if their inverse cumulative function is easily computable. This method is used to simulate the exponential distribution ($\varepsilon(\lambda)$).

Normal distribution: Box Muller transform Sometimes, inverting a cumulative function is difficult. In these cases, special algorithms are used. For example, a normal distribution ($\mathcal{N}(\mu,\sigma)$) does not have a known inverse function, so previous method is useless. However, many other methods exist to

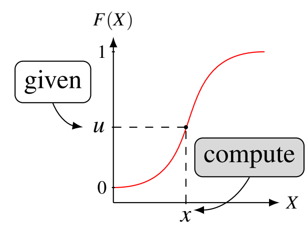


Fig. 4. Simulation of random values x given a cumulative function $F(X)$ of a random variable X and uniform u

simulate a normally distributed sample. Our implementation uses Box Muller algorithm.

For a more complete overview about probability and simulation, please refer to [25].

This section described the method we use to improve the meaningfulness of generated models. We exploit domain-specific quality metrics, and the simulation probability distributions to achieve our goal.

V. Case Studies

This section experimentally shows that using probability distributions improves the meaningfulness of generated datasets. We consider two case studies, one from Software Engineering area and the other from Bioinformatics.

A. *Java code generation*

One of the main objectives of our approach is the generation of benchmarks of test programs for different applications, such as compilers or virtual machines. In this experiment, we generate realistic and relevant skeletons of Java programs using real code measurements. We choose Java for facility to find real programs to collect desired measurements. However, our method can be applied to any programming language.

We collected 200 real Java projects coming from two corpus (Github and Qualitas corpus [26]. For more heterogeneity, we

TABLE I. Chosen code metrics with their theoretical probability distribution. ε: Exponential distribution, $\mathcal{N}$: Normal distribution.

Metric	$\rightsquigarrow$	**Theoretical distrib.**
Class/Package	$\rightsquigarrow$	$\varepsilon(\frac{1}{8.387})$
Methods/Type	$\rightsquigarrow$	$\varepsilon(\frac{1}{7.06})$
Attributes/Type	$\rightsquigarrow$	$\mathcal{N}(3.46, 2.09)$
Constructor/Type	$\rightsquigarrow$	$\mathcal{N}(0.73, 0.26)$
Sub-Classes/Classe	$\rightsquigarrow$	$\varepsilon(\frac{1}{0.22})$
% Interface/Package	$\rightsquigarrow$	$\varepsilon(\frac{1}{8.001})$
Parameters/Methods	$\rightsquigarrow$	$\mathcal{N}(0.87, 0.25)$

chose projects having different sizes (big project for qualitas corpus and smaller ones from github) and different origins (well-known software such as Eclipse, Apache or ArgoUML and also small software written by only one developer). We measured metrics related to their structure, such as the percentage of concrete classes/ abstract classes, the average number of constructors for a class, the visibility of fields and methods, etc [23]. To measure these metrics we used an open source tool called `Metrics` [27]. After that, we use R software [28] to compute histogram of each metric in order to deduce its theoretical probability distribution. Table I gives the different metrics and their theoretical probability distributions. Figures 5 and 6 show two examples of real corpus histograms.

According to these metrics, we automatically generate Java programs having the same characteristics as the real ones. To achieve this goal, we design a meta-model representing skeletons of Java projects and we adjoin some OCL constraints (about 10 constraints have been added to the Java meta-model we designed). For example:

- All classes inside a package have a different name.
- A class cannot extend itself.

Finally, 300 Java projects are generated using three versions of our approach. Then, we obtain four corpus:

1) Projects generated by $\mathcal{G}$RIMM but without OCL.
2) Projects generated by $\mathcal{G}$RIMM plus OCL.
3) Projects generated by $\mathcal{G}$RIMM plus OCL and probability distributions (denoted $\mathcal{G}$ЯRIMM in figures).
4) Real Java projects.

We compare, for these four cases, the distributions of constructors per class and the visibility of attributes in Figures 5 and 6.

We observe that the two first versions without probability distributions give results that are very far from the characteristics of real models. On the other hand, introducing simulated probability distributions leads to substantial improvement. We see that the distribution of the number of constructors and the visibility of attributes (Figures 5,6) of generated models are close to real ones. Moreover, these results are always better when adding probabilities for all other measurements presented in Table I.

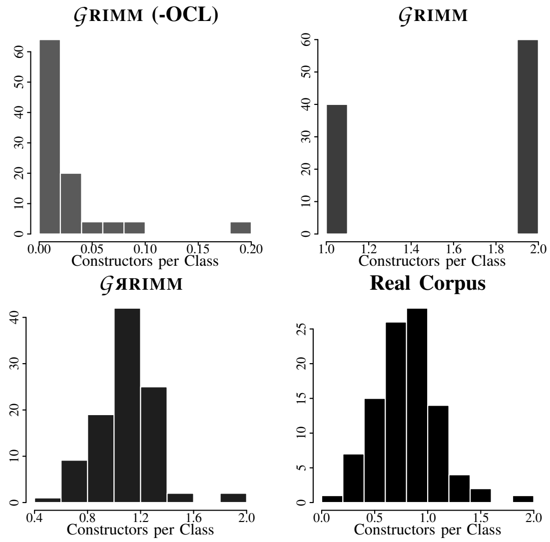


Fig. 5. Comparing the number of constructors per class in Java projects.

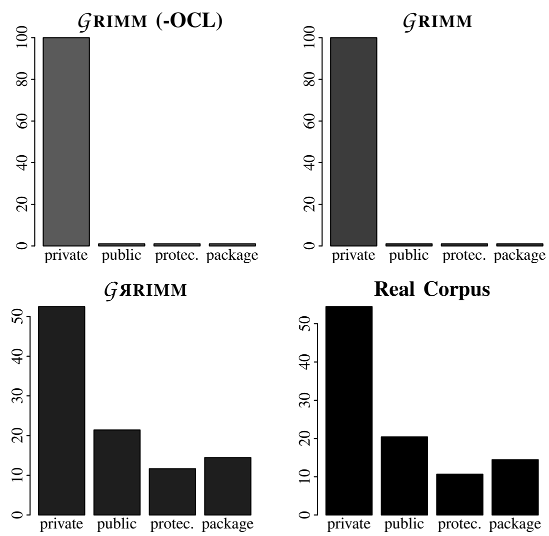


Fig. 6. Comparing the visibility of attributes in Java projects.

Despite these encouraging results, some threats to the validity of this experiments can be given. A first one is a construction threat related to the chosen metrics. Indeed, a lot of metrics can be found in the literature. In our experiment we choose only those related to the structure of a project. There is a second internal threat concerning real Java corpus. They may not be sufficiently representative of all the Java projects even if they come from a well-known corpus and a famous repository.

B. Scaffold graphs generation

Scaffold graphs are used in Bioinformatics to assist the reconstruction of genomic sequences. They are introduced late in the process, when some DNA sequences of various lengths, called *contigs*, have already been produced by the assembly step. Scaffolding consists in ordering and orienting the contigs, thanks to oriented relationships inferred from the initial sequencing data. A scaffold graph is built as follows: vertices represent extremities of the contigs, and there are two kind of edges. Contig edges link both extremities of a given contig (strong edges in Figure 8), whereas scaffolding edges represent the relationship between the extremities of distinct contigs. Contig edges constitute a perfect matching in the graph, and scaffolding edges are weighted by a confidence measure. Those graphs are described and used in the scaffolding process in [29] for instance. The scaffold problem can be viewed as an optimisation problem in those graphs, and consists in exhibiting a linear sub-graph from the original graph. Therefore, it can be considered as well as a model transformation, when models conform to the Scaffold graph meta-model given in Figure 7, that we designed. Producing datasets to test the algorithms is a long process, somehow biased by the choices of the methods (DNA sequences generation, assembly, mapping), and there does not exist a benchmark of scaffold graphs of various sizes and densities. Moreover, real graphs are difficult and expensive to obtain. Thus, it is interesting to automatically produce scaffold graphs of arbitrary sizes, with characteristics close to the usual ones. In [29], the authors present some of these characteristics, that are used here to compare the GRIMM instances vs. the "hand-made" graphs.

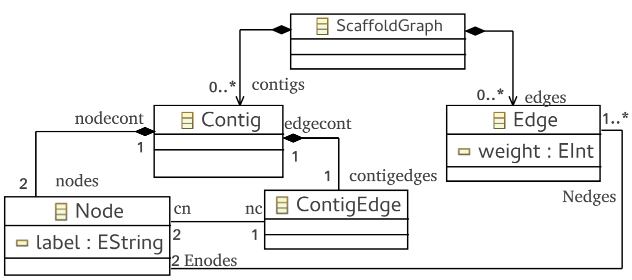


Fig. 7. A meta-model for the scaffold graphs.

The probability distribution chosen to produce the graph emerges from the observation that the degree distribution in those graphs is not uniform, but follows an exponential distribution. We compare several datasets, distributed in several classes according to their sizes:

1) Graphs generated by GRIMM plus OCL.
2) Graphs produced by GRIMM plus OCL and probability distributions (denoted GЯRIMM in figures).
3) Real graphs of different species, described in [29].

For each real graph, 60 graphs of the same size are automatically generated. 30 graphs are naively generated using the original GRIMM method [8], [2], and 30 others are generated after the simulating of probability distribution. These models are then compared in term of visual appearance (Figure 8), degree distribution (Figure 9) and according to some graph measurements (Table II).

We see in Figure 8 three models (scaffolds graphs) corresponding to the same species (monarch butterfly). More

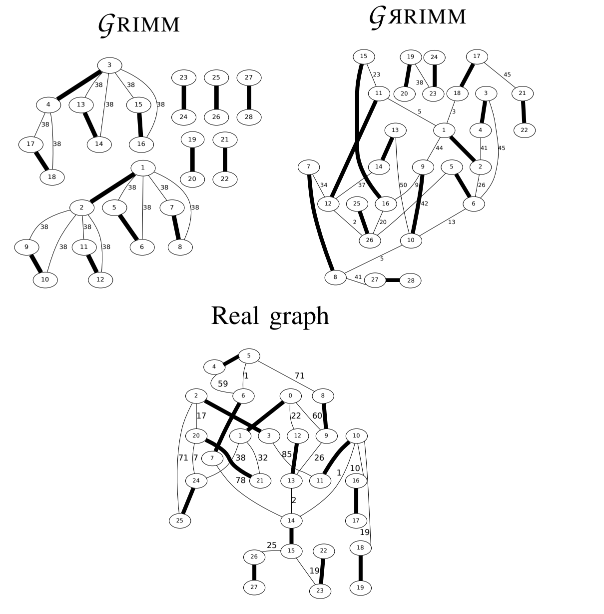


Fig. 8. Three Scaffold graphs corresponding to the same species (monarch butterfly). Strong edges represent contig edges, other edges are scaffolding edges.

precisely, it refers to mitochondrial DNA of monarch butterfly. The naive method generates a graph that does not look like the real one. This graph is too weakly connected, and the connected parts have a recurring pattern. This is not suitable for a useful scaffold graph. Whereas, introducing probabilities provides graphs having shapes close to reality. Thus, both real graphs and generated graphs (with probability distribution) are strongly connected, and more randomness can be observed in the connections and the weights of edges.

Figure 9 compares the degree distributions for three scaffold graphs of the same species. We see that generating with probabilities gives a distribution very similar to the distribution in the real graph.

Table II compares the three benchmarks of scaffold graphs (naive generation, generation with probabilities and real graphs) according to some graph measurements. We can observe again, that the graphs generated with probabilities are closer to real graphs than the naively generated graphs in all cases. The measurements on the naive graph suffer from a lack of diversity and randomness. Indeed, the minimal and the maximal degree are the same for all generated models. This, of course, does not reflects the reality. Notice also that it was not possible with the naive generation method to generate largest graphs corresponding to largest genomes.

This section presented two different case studies, to show how we exploit domain-specific metrics, for improving meaningfulness of generated models. The following sections are devoted to the second aspect of our contributions. It describes the approach we developed to increase inter-model diversity.

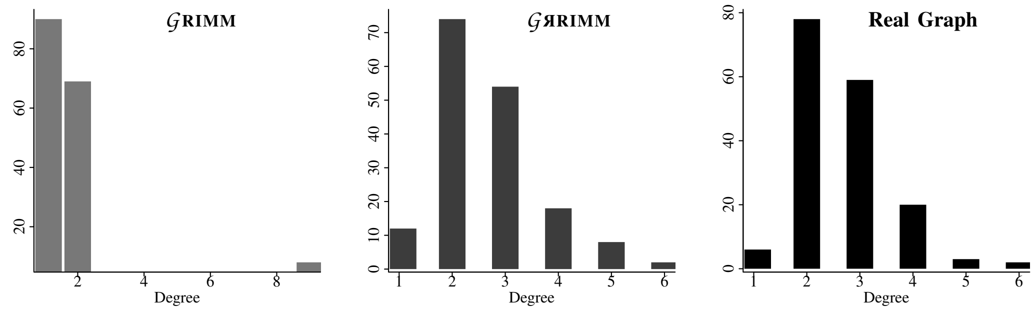


Fig. 9. Comparing the degree distribution between a real scaffold graph and its equivalent generated ones (168 nodes and 223 edges).

TABLE II. Comparing graph metrics on real scaffold graphs and average for 60 generated ones for each species.

Graph size			Measurements					
			GRIMM generation		**GЯRIMM generation**		**Real graphs**	
Species	nodes	edges	min/max degree	h-index	min/max degree	h-index	min/max degree	h-index
monarch	28	33	1/9	3	1/ 4.6	4.06	1/ 4	4
ebola	34	43	1/9	3	1/ 4.83	4.60	1/ 5	4
rice	168	223	1/9	8	1/ 6.03	5.93	1/ 6	5
sacchr3	592	823	1/9	10	1/ 7	6.76	1/ 7	6
sacchr12	1778	2411	–	–	1/ 7.53	7	1/10	7
lactobacillus	3796	5233	–	–	1/ 8.06	7.8	1/12	8
pandora	4092	6722	–	–	1/ 8.23	7.96	1/ 7	7
anthrax	8110	11013	–	–	1/ 8.3	8.03	1/ 7	7
gloeobacter	9034	12402	–	–	1/ 8.46	8	1/12	8
pseudomonas	10496	14334	–	–	1/ 8.43	8	1/ 9	8
anopheles	84090	113497	–	–	1/ 8.96	9	1/ 51	12

VI. Measuring model differences

Brun and Pierantonio state in [30] that the complex problem of determining model differences can be separated into three steps: *calculation* (finding an algorithm to compare two models), *representation* (result of the computation being represented in manipulable form) and *visualization* (result of the computation being human-viewable).

Our comparison method aims to provide solutions to compare not only two models between them but a whole set of models or sets of models. The rest of this section describes in details the calculation algorithms we choose to measure model differences. Since our method aims to compare sets of models, we took care to find the quickest algorithms. Because chosen comparison algorithms are called hundreds of time to manipulate one set containing dozens of models.

As a proof of concept, we consider here four different distances to express the pairwise dissimilarity between models. As stated in [31], there is intrinsically a difficulty for model metrics to capture the semantics of models. However, formalizing metrics over the graph structure of models is easy, and they propose ten metrics using a multidimensional graph, where the multidimensionality intends to partially take care of semantics on references. They explore the ability of those metrics to characterize different domains using models. In our work, we focus on the ability of distances to seclude models inside a set of models. Thus, we have selected very various distances, essentially of 2 different area: distances on words (from data mining and natural language processing) and distances on graphs (from semantic web and graph theory). Word distances have the very advantage of a quick computation, whereas graph distances are closer to the graph structure of models. As already said, an interesting feature is the fact that all those distances are, in purpose, not domain-specific, not especially coming from MDE, but adapted to the latter.

A. *Words distances for models*

We define two distances for models based on distances on words: the hamming distance and the cosine distance. The first one is really close to syntax and count the number of difference between two vectors. The second one is normalized and intends to capture the multidimensional divergence between two vectors representing geometrical positions.

1) From models to words: We define the vectorial representation of a model as the vector collecting links and attributes' values of each class instance, as illustrated on the model of Figure 10. At the left-hand-side of the figure is an example of meta-model. At the right-hand-side of the figure are two models conform to this meta-model, and their vectorial representation. The obtained vector from a model *m* is composed of successive sections of data on each instance of *m*, when data is available. Each section of data is organized as follows: first data on links, then data on attributes. When there is no such data for a given instance, it is not represented. In the example of Figure 10, instances of *B*, which have

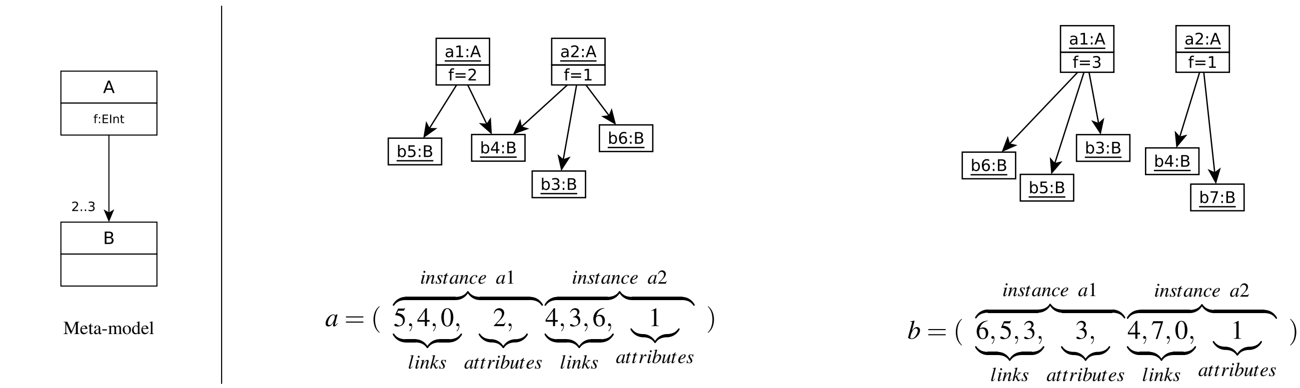


Fig. 10. Two small models and their vectorial representation.

no references and no attributes, as imposed by the meta-model, are not directly represented in the vectors. However, they appear through the links attached to instances of A. An attribute is represented by its value. A link from an instance i to an instance j is represented by the number of the referenced instance j. Each instance of a given meta-class mc, are represented by sections of identical size. Indeed, all the instances of mc have the same number of attributes. The number of links may vary from an instance to another, but a size corresponding to maximal cardinality is systematically attributed. This cardinality is either found in the meta-model or given in the generation parameters. When the actual number of links is smaller than the maximal number of links, 0 values are inserted.

2) Hamming distance for models: Hamming distance compares two vectors. It was introduced by *Richard Hamming* in 1952 [32] and was originally used for fault detection and code correction. Hamming distance counts the number of differing coefficients between two vectors.

The models to compare are transformed into vectors, then we compare the coefficients of vectors to find the distance between both models:

a	=	(5,	4,	0,	2,	4,	3,	6,	1)
b	=	(6,	5,	3,	3,	4,	7,	0,	1)
d(a,b)	=	1+	1+	1+	1+	0+	1+	1+	0
	=	$\frac{6}{8}$							

Richard Hamming's original distance formula is not able to detect permutations of links, which leads to artificially higher values than expected. In our version, we sort the vectors such as to check if each link exists in the other vector. In the previous example, the final distance then equals to $\frac{5}{8}$. The complexity is linear in the size of models, due to the vectorization step. Notice also that this distance implies that vectors have equal sizes. This is guaranteed by the way we build those vectors.

3) Cosine distance: Cosine similarity is a geometric measure of similarity between two vectors, ranging from -1 to 1: two similar vectors have a similarity that equals 1 and two diametrically opposite vectors have a cosine similarity of -1. Cosine similarity of two vectors a and b is given by the following formula:

$$C_S(a,b) = \frac{a.b}{||a||.||b||} = \frac{\sum_{i=1}^{n} a_i.b_i}{\sqrt{\sum_{i=1}^{n} a_i^2}.\sqrt{\sum_{i=1}^{n} b_i^2}}$$

After a vectorization of models, cosine similarity is then used to compute a normalized cosine distance over two vectors [33]:

$$C_D(a,b) = \frac{1 - C_S(a,b)}{2}$$

Again, the time complexity of the computation is linear in the size of models.

4) Levenshtein distance for models: Levenshtein distance [34] (*named after Vladimir Levenshtein*) is a string metric used to compare two sequences of characters. To summarise the original idea, a comparison algorithm counts the minimal number of single-character edits needed to jump from a first string to a second one. There exist three character edit operations: addition, deletion and substitution.

Our customized Levenshtein distance is based on the vectorial representation of a model. Each character in original distance is replaced by a class instance of the model. So, we count the minimal cost of class instance edit operations (addition, deletion or substitution) to jump from the first model to the second one.

First, a vectorial representation of a model is created according to the class diagram given in Figure 11. Then, we determine the cost of each one of the three edit operations over instanceOfClass objects. `instanceCost` method gives the cost to add or to delete an instanceOfClass. It counts the number of edges and the number of attributes of this instance. `substituCost` method gives the cost to substitute an instance by another one. To determine the substitution cost,

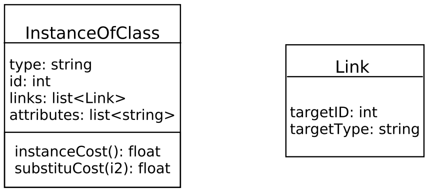


Fig. 11. Class diagram for instanceOfClass and Link to build a vectorial representation of a model.

we count the number of common links and attributes. Thus, two `instanceOfClass` are exactly equal if they have the same type, their links have the same type and all their attributes have the same values.

Finally, Levenshtein algorithm [34] is applied and a metric of comparison is computed. Our comparison metric gives the percentage of common elements between two models.

B. Centrality distance for models

Centrality is a real function that associates a value to each node of a graph [35]. This value indicates how much a node is *central* in this graph, according to a chosen criterion. For example, in a tree, the highest value of centrality is given to the root of the tree, whereas the smallest values are associated to the leaves. A centrality function C is defined by:

$$C: \quad E \to \mathbb{R}^+ \\ v \mapsto C(v)$$

Many usual centrality functions exist. The simplest one, the *degree centrality*, associates to each node its degree. Among the well-known centrality functions, we can cite: betweenness centrality, closeness centrality, harmonic centrality, etc.

In this paper, we propose a novel centrality function adapted for MDE and based on *eigenvector* centrality. This centrality was also used in the first published version of `PageRank` algorithm of the `Google` search engine [36]. In `PageRank`, eigenvector centrality is used to rank the web pages taken as nodes of the same graph.

1) From models to graphs: Centrality functions are defined on graphs, and models could be considered as labelled and typed graphs. Our graph representation of models is obtained as follows:

- Create a node for each class instance (central nodes).
- Create a node for each attribute (leaf nodes).
- Create an edge from each class instance to its attributes.
- Create an edge for each simple reference between two class instances.
- Create two edges if two class instances are related by two opposite references.
- Create an edge for each composition link.

Tables III and IV summarize and illustrate these transformations rules. Real numbers c, r and t represent the weights assigned to composition links, reference links and attributes.

TABLE III. NODES TRANSFORMATION RULES.

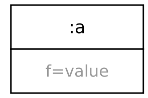
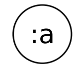
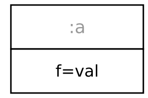
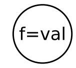
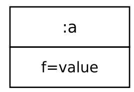
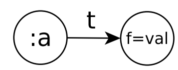
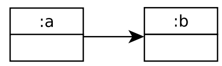
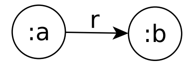
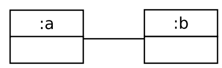
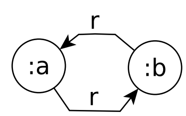
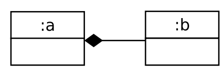
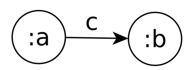
Model element	Graph element
	
	

TABLE IV. EDGES TRANSFORMATION RULES.

Model element	Graph element
	
	
	
	

2) Centrality measure: Our centrality is inspired from `pagerank` centrality and adapted to models, taking into account class instances and their attributes, links between classes (input and output) and types of link between two classes (simple references, two opposite references or compositions). For a given node v of the graph, we denote by $N(v)$ the set of its neighbors. The following function gives the centrality of each node v:

$$C(v) = \sum_{u \in N^+(v)} \frac{C(u)}{deg(u)} \times w(v,u).$$

$w(v,u)$ gives the weight of the link between node v and u, determined by the kind of link between them (attribute, reference or composition). The weight of a link can be given by the user or deduced from domain-based quality metrics. For instance, *Kollmann and Gogolla* [37] described a method for creating weighted graphs for UML diagrams using object-oriented metrics.

3) Centrality vector: The centrality vector C contains the values of centrality for each node. The previous centrality function induces the creation of a system of n variables equations: $C(v_i) = c_1 C(v_1) + c_2 C(v_2) + \ldots + c_i C(v_i) + \ldots + c_n C(v_n)$.

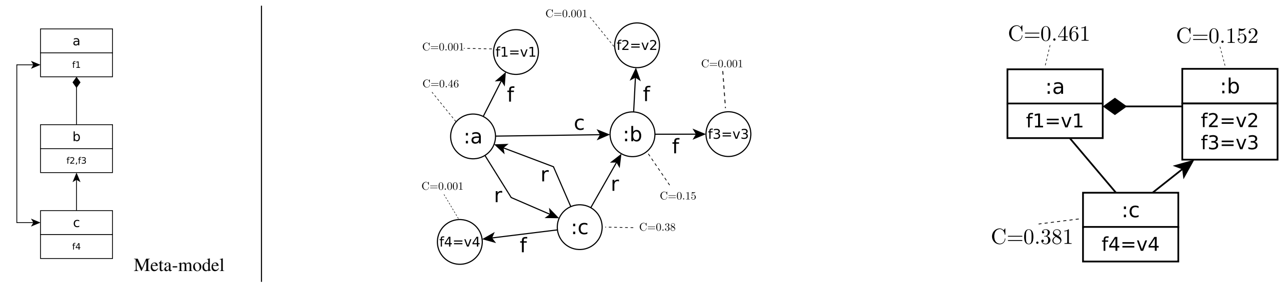


Fig. 12. Centrality vector computed for an example model and its equivalent graph.

To compute the centrality vector C we must find the eigenvector of a matrix A whose values are the coefficients of the previous equations: $C = AC$, where A is built as follows:

$$A_{ij} = \begin{cases} 0 \text{ if } (v_i, v_j) \notin \text{ Graph}, \\ \dfrac{w(v_i, v_j)}{N(v_i)} \text{ otherwise.} \end{cases}$$

After building matrix A, we use the classical algorithm of power iteration (also known as Richard Von Mises method [38]) to compute the centrality vector C.

The result centrality vector has a high dimension (see example on Figure 12). To reduce this dimension therefore improve the computation's efficiency, we group its coefficients according to the classes of the meta-model. Then the dimension equals to the number of classes in the meta-model.

4) Centrality distance: Roy et al. proved in [39] that a centrality measure can be used to create a graph distance. Here, the centrality vectors C_A and C_B of two models A and B are compared using any mathematical norm: $d(A,B) = ||C_A - C_B||$.

C. Discussion

We use in previous paragraphs representations of models, which could be discussed. Indeed, there are potentially many ways to vectorize models, and we choose one highly compatible with our tool. Since CSP generation already provides a list of classes attributes and links, we simply used this representation as entry for the metrics. Again, transforming models into graphs and trees may be done through several ways. We arbitrarily choose one way that seemed to capture the graph structure. Our goal here, was to test different and diversified manners to represent a model and proposed some distance between them, not to make an exhaustive comparison study between quality of representation versus metrics. This study will be done in future works.

VII. Handling sets of models

In this section, we propose an automated process for handling model sets. The purpose is to provide solutions for comparing models belonging to a set, selecting the most representative models in a set and bringing a graphical view of the concept of diversity in a model set.

This process helps a user in choosing a reasonable amount of models to perform his experiments (e.g., testing a model transformation). Moreover, using our approach, the chosen model set aims to achieve a good coverage of the meta-model's solutions space.

If there are no available models, a first set of models is generated using GЯIMM tool [9]. These generated models are conform to an input meta-model and respect its OCL constraints. When probability distributions related to domain-specific metrics are added to the process, intra-model diversity is improved. Our goal is to check the coverage of the meta-model's solutions space. In other words, we want to help a user to answer these questions: (1) how to quantify the inter-model diversity? (2) Are all these models useful and representative? (3) Which one of my model sets is the most diverse?

A. Comparison of model sets

Distance metrics proposed in Section VI compare two models. To compare a set of models, we have to compute pairwise distances between models inside the set. A symmetrical distance matrix is then created and used to quantify the inter-model diversity. It is noticeable that, thanks to the modularity of the approach, this step can be replaced by any kind of dataset production. For instance, if the user already has a set of models, it is possible to use it instead of the generated one. Moreover, another distance metric can be used instead of the metrics we propose.

B. Selecting most representative models

Our idea is that when a user owns a certain number of models (real ones or generated ones), there are some of them, which are representative. Only these models should be used in some kind of tests (e.g., robustness or performance). Most of other models are close to these representative models.

We use *Hierarchical Matrix clustering* techniques to select the most representative models among a set of models. The distance matrix is clustered and our tool chooses a certain number of models. In our tool, we use the hierarchical clustering algorithm [40], implemented in the R software (*hclust, stats package, version 3.4.0*) [41]. This algorithm starts by finding a tree of clusters for the selected distance matrix as shown in Figure 13. Then, the user has to give a threshold value in order to find the appropriate value. This value depends on the diversity the user wants. For example, if the user wants models sharing only 10% of common elements, then 90% is the appropriate threshold value. This value depends also on the used metric. Thus, Levenshtein distance compares the names of elements and the values of attributes, leading to choose a smaller threshold value (for the same model set) than for centrality distance, which compares only the graph structure of the models.

Using the clusters tree and the threshold value, it is easy to derive the clusters, by cutting the tree at the appropriate height (Figure 13). The most representative models are chosen by arbitrarily picking up one model per cluster. For instance, 3 different clusters are found using the tree of clusters in Figure 13. Clone detection can also be performed using our approach by choosing the appropriate threshold value. Indeed, if threshold equals to 0, clusters will contain only clones.

TABLE V. AN EXAMPLE OF DISTANCE MATRIX (HAMMING) FOR 10 MODELS.

	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}
m_1	0	12	27	27	27	26	46	44	45	39
m_2	12	0	27	26	27	27	45	45	43	40
m_3	27	27	0	18	17	16	46	45	46	39
m_4	27	26	18	0	18	18	45	44	45	40
m_5	27	27	17	18	0	18	45	43	44	38
m_6	26	27	16	18	18	0	45	44	46	40
m_7	46	45	46	45	45	45	0	36	36	41
m_8	44	45	45	44	43	44	36	0	34	37
m_9	45	43	46	45	44	46	36	34	0	39
m_{10}	39	40	39	40	38	40	41	37	39	0

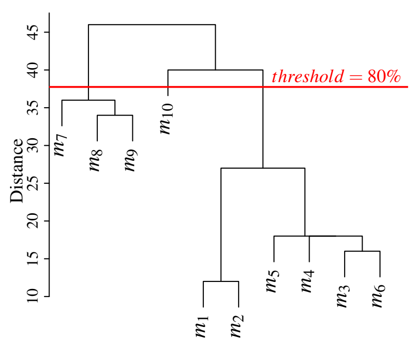


Fig. 13. Clustering tree computed form matrix in Table V.

C. Graphical view of diversity

Estimating diversity of model sets is interesting for model users. It may give an estimation on the number of models needed for their tests or experiments and they can use this diversity measure to compare between two sets of models.

When the number of models in a set is small, diversity can be done manually by checking the distance matrix. Unfortunately, it becomes infeasible when the set contains more than an handful of models. We propose a human-readable graphical representation of diversity and solutions' space coverage for a set of models.

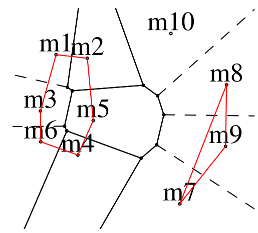


Fig. 14. Voronoi diagram for 10 models compared using Hamming distance.

Our tool creates *Voronoi tessellations* [42] of the distance matrix in order to assist users in estimating the diversity or in comparing two model sets. A *Voronoi* diagram is a 2D representation of elements according to a comparison criterion, here distances metrics between models. It faithfully reproduces the coverage of meta-model's solutions space by the set of models. Figure 14 shows the Voronoi diagram created for the matrix in Table V. The three clusters found in the previous step are highlighted by red lines. We use the Voronoi functions of R software (available in package *tripack, v1.3-8*).

VIII. TOOLING

This section details the tooling implementing our contributions. All the algorithms and tools are in free access and available on our web pages [43].

Our tool for comparing models and handling model sets is called COMODI (COunting MOdel DIfferences). It consists in two different parts. The first one, written in *java*, is used to measure differences between two models using the above 4 metrics. The second part, written in *bash* and *R*, provides algorithms for handling model sets (comparison, diversity estimation and clustering).

A. Comparing two models

It is possible to measure the differences between two models using COMODI. For that you just need to give as input two models and their *ecore* meta-model. Out tool supports two different formats: *dot* model files produced by GRIMM and *xmi* model files. COMODI supports all xmi files produced by EMF of generated by GRIMM, *EMF2CSP* or PRAMANA tools.

The first step is to parse the input models into the appropriate representation (graph or vector). Then, the above distance algorithms are applied. COMODI outputs different model comparison metrics in command line mode. Process of COMODI is described in Figure 15.

B. Handling a set of models

Our tool is also able to handle sets of models and produce distance matrices, perform clustering and plot diagrams and

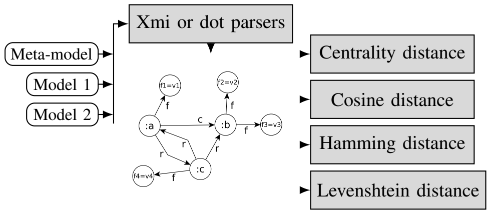


Fig. 15. Comparing two models using COMODI tool.

give some statistics. The input of the tool is a folder containing the models to compare and their *ecore* meta-model. The supported formats for models are the same as described above (xmi and dot).

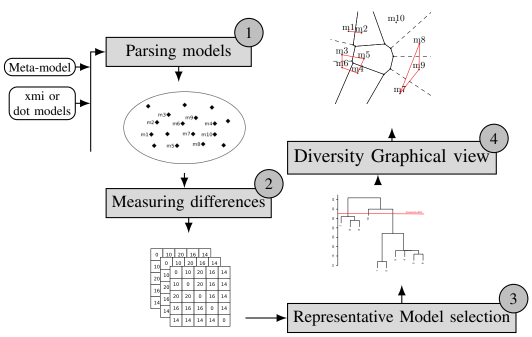


Fig. 16. Handling a set of models using COMODI tool.

After parsing all the models into the appropriate representation for each metric, distance matrices are produced by pairwise comparison of models. *R* scripts are called to perform hierarchical clustering on these matrices. This allows us to select the most representative models of that folder. Voronoi diagrams are plotted and can be used to estimate the coverage of the folder and to compare the diversity of two folders. COMODI prints also some simple statistics on models: closest models, most different models, etc. These steps are shown in Figure 16.

IX. Application: Improving diversity

The main contributions of this paper - distances between models, representative model selection to improve diversity - were used in a work in bioinformatics (named scaffolding) [44]. A genetic approach is paired with GЯIMM model generation tool to improve the diversity of a set of automatically generated models. Figure 17 shows how we start from a GRIMM model set (left) with few difference between them, to GЯIMM (center) with a better distribution due to the probability sampler, to something very relevant by using a genetic approach (right) based on these model distances in order to improve diversity.

We want to address the following question: do proposed distances and process of automated models selection help to improve the diversity and the coverage of generated models. We chose one meta-model (Figure 7) modeling a type of graphs involved in the production of whole genomes from new-generation sequencing data [29]. Hereinafter, we give the experimental protocol:

- Generate 100 initial models conforming to the scaffold graph meta-model using GЯIMM tool [9].
- Model the problem of improving diversity using genetic algorithms (GA). Our modeling in GA can be found in [11].
- Run 500 times the genetic algorithm. At each step, use model distances and automatic model selection to choose only the best models for the next step.
- View final results in terms of model distances and meta-model coverage using Voronoi diagrams.

The whole process induces the creation of up to $50{,}000$ different models. Each following figures required about 3h CP to be computed.

Curves on Figure 18 show the evolution of hamming and cosine distance while the genetic algorithm is running (minimum, maximum and mean distance over the population at each generation). We can observe that both cosine distance and hamming distance help to improve diversity of generated models. The quick convergence of both curves (around 100 iterations of GA) is a good way to check the efficiency of both models distances. We observe that the worst case in the final population is better than the best case in the initial population, thus we reached a diversity level that we did not obtained in the initial population obtained with GЯIMM.

We introduce several improvements to describe the fitness function used in genetic selection [11] and improve median value for final population from 0.7 up to 0.9 for Hamming and from 0.11 to 0.15 for maximum with Cosinus distance. Figure 19 compares the models produced by the different distances. Red (resp. blue) dotplots represent the distribution of distances on the final population computed using Hamming distance (resp. Cosine distance). On the left, models are compared using Hamming distance, on the right, they are compared using Cosine distance. We remark that different distances do not produce the same final models. Indeed, we can observe that the best selected models for Hamming distance obtain lower scores when compared using Cosine distance, and vice versa. Other experimental results show that our four model distances can be used in a multi-objective genetic algorithm since they treat different constructions of the meta-model. Results are better on the final model set in terms of diversity and coverage, than when only one kind of distance is used.

Figure 20 shows two Voronoi diagrams of 100 models. The first one is computed on the initial set of models, the second on the set of models generated after the 500^{th} iteration of the genetic algorithm. We kept the same scale to visualize the introduced seclusion. Here we can see the insufficient solutions' space coverage of the first Voronoi diagram. After running the multi-objective genetic algorithm, we observe a better coverage of the space. At the end of the process, we obtain 100 very distinct models.

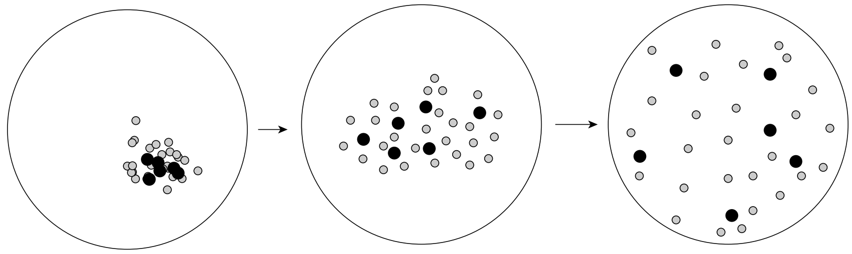


Fig. 17. Diversity improving process. Black circles are the most representative models of the set.

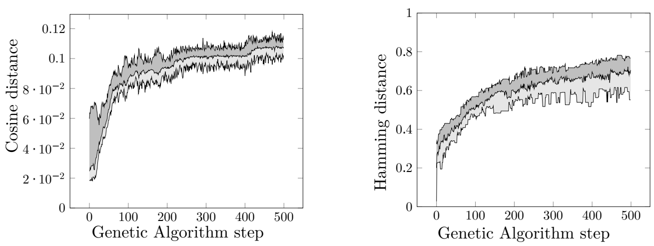


Fig. 18. Minimum, average and maximum hamming and cosine distance while running the genetic algorithm.

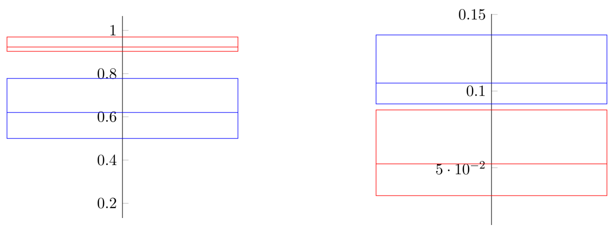


Fig. 19. Comparison of best selected models pairwise distances distributions.

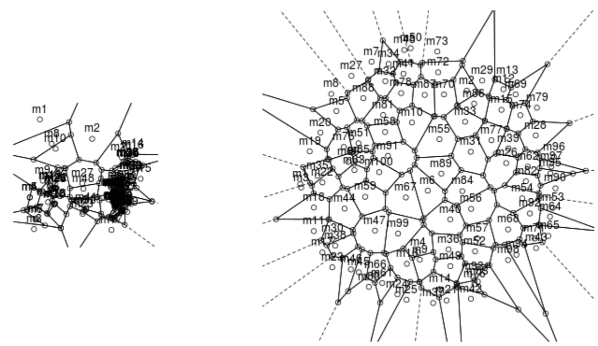


Fig. 20. Solutions' space coverage of the initial set of models (left) compared to the last iteration (500^{th}) of the genetic algorithm (right).

X. Conclusion

This paper gathers contributions developed, in order to complete a project (tool and methodology) named $\mathcal{G}$RIMM ($\mathcal{G}$eneRating Instances of Meta-Models). The topic of these contributions is improving the inherent quality of automatically generated models. We investigated the relation between the *quality* of the generated models and the concept of *diversity*. Our claim is that increasing the diversity when generating models, will improve their quality and usefulness, because generated models are becoming closer to real models.

We distinguish two different aspects of diversity, which are solved separately: *intra-model diversity* and *inter-model diversity*.

A. Intra-model diversity

The goal of intra-model diversity is to enrich the diversity inside a model. It is obtained by injecting randomness during the generation process. Element naming, linking objects become less *systematic* and *naive*. Obtained models are closer to real models, i.e., meaningful. We propose to use probability distributions, that are related to domain-specific quality metrics. Parameters for probability laws can be chosen by users, but they are preferably gathered by a statistical study of real data. Domain knowledge is also used when real data is rare.

We observed a substantial improvement of meaningfulness when simulated probabilities samples are added to the model generator. New generated instances have characteristics very close to real models, improving their usefulness for testing programs. This is especially interesting when data is rare, difficult or expensive to obtain, as for scaffold graphs.

B. Inter-model diversity

Inter-model diversity tries to maximize the diversity of a set of models. The idea is to cover as much area in solutions' space as possible, in order to get the best diversity ever. In our work, we propose to use model comparison and then genetic algorithm to achieve this goal.

Counting model differences is a recurrent problem in Model Driven Engineering, mainly when sets of models have to be compared. This paper tackles the issue of comparing two models using several kinds of distance metrics inspired from distances on words and distances on graphs. An approach and a tool are proposed to handle sets of models. Distance metrics are applied to those sets. Pair of models are compared and a matrix is produced. We use hierarchical clustering algorithms to gather the closest models and put them in subsets. Our tool, COMODI, is also able to choose the most representative models of a set and give some statistics on a set of models. Human readable graphical views are also generated to help users in doing that selection manually.

First, sets of non-diverse models are automatically generated using, for example, $\mathcal{G}$RIMM tool. Then, COMODI is coupled to a genetic algorithm to improve the diversity of this first set of models.

C. Future work

One interesting future improvement to $\mathcal{G}$RIMM is to become a tool for assisting model designers, by giving quick and graphically viewable feedback during the design process. At the moment, it generates models that are close to real ones and graphical visualization is possible. Nevertheless, error detection is still poor, and it is a restriction for a large diffusion. Currently, we work on improving the interaction between the tool and the user. New features are added to the process of model generation in order to detect instantiation failures, and to correct them, or to help the user to do the correction. Guiding the user is done by interpreting the output of the *CSP solver* to understand the origin of any faced problem.

D. Application

The problematic of handling sets of models and the notion of distance is also involved in many other works related to testing model transformations. All these issues are interesting applications to the contributions of this paper. For example, *Mottu et al.* in [45] describe a method for discovering model transformations pre-conditions, by generating test models. A first set of test models is automatically generated, and used to execute a model transformation. Excerpts of models that make the model transformation failing are extracted. An expert then tries manually and iteratively to discover pre-conditions using these excerpts. Our common work aims to help the expert by reducing the number of models excerpts and the number of iterations to discover most of pre-conditions. A set of models excerpts is handled using COMODI and clusters of close models are generated. Using our method, the expert can find many pre-conditions in one iteration and using less model excerpts.

REFERENCES

[1] A. Ferdjoukh, F. Galinier, E. Bourreau, A. Chateau, and C. Nebut, "Measuring Differences To Compare Sets of Models and Improve Diversity in MDE," in *ICSEA, International Conference on Software Engineering Advances*, 2017, pp. 73–81.

[2] A. Ferdjoukh, A.-E. Baert, A. Chateau, R. Coletta, and C. Nebut, "A CSP Approach for Metamodel Instantiation," in *IEEE ICTAI*, 2013, pp. 1044–1051.

[3] S. Sen, B. Baudry, and J.-M. Mottu, "Automatic Model Generation Strategies for Model Transformation Testing," in *ICMT, International Conference on Model Transformation*, 2009, pp. 148–164.

[4] C. A. González Pérez, F. Buettner, R. Clarisó, and J. Cabot, "EMFtoCSP: A Tool for the Lightweight Verification of EMF Models," in *FormSERA, Formal Methods in Software Engineering*, 2012, pp. 44–50.

[5] F. Hilken, M. Gogolla, L. Burgueño, and A. Vallecillo, "Testing models and model transformations using classifying terms," *Software & Systems Modeling*, pp. 1–28, 2016.

[6] D. S. Kolovos, D. Di Ruscio, A. Pierantonio, and R. F. Paige, "Different models for model matching: An analysis of approaches to support model differencing," in *CVSM@ICSE*, 2009, pp. 1–6.

[7] D. Steinberg, F. Budinsky, M. Paternostro, and E. Merks, *EMF: Eclipse Modeling Framework 2.0*. Addison-Wesley Professional, 2009.

[8] A. Ferdjoukh, A.-E. Baert, E. Bourreau, A. Chateau, and C. Nebut, "Instantiation of Meta-models Constrained with OCL: a CSP Approach," in *MODELSWARD*, 2015, pp. 213–222.

[9] A. Ferdjoukh, E. Bourreau, A. Chateau, and C. Nebut, "A Model-Driven Approach to Generate Relevant and Realistic Datasets," in *SEKE*, 2016, pp. 105–109.

[10] F. Rossi, P. Van Beek, and T. Walsh, Eds., *Handbook of Constraint Programming*. Elsevier Science Publishers, 2006.

[11] F. Galinier, E. Bourreau, A. Chateau, A. Ferdjoukh, and C. Nebut, "Genetic Algorithm to Improve Diversity in MDE," in *META*, 2016, pp. 170–173.

[12] J. Cabot, R. Clarisó, and D. Riera, "Verification of UML/OCL Class Diagrams using Constraint Programming," in *ICSTW, IEEE International Conference on Software Testing Verification and Validation Workshop*, 2008, pp. 73–80.

[13] A. Mougenot, A. Darrasse, X. Blanc, and M. Soria, "Uniform Random Generation of Huge Metamodel Instances," in *ECMDA, European Conference on Model-Driven Architecture Foundations and Applications*, 2009, pp. 130–145.

[14] H. Wu, "An SMT-based Approach for Generating Coverage Oriented Metamodel Instances," *IJISMD, International Journal of Information System Modeling and Design*, vol. 7, no. 3, pp. 23–50, 2016.

[15] K. Ehrig, J. Küster, and G. Taentzer, "Generating Instance Models from Meta models," *SoSyM, Software and Systems Modeling*, pp. 479–500, 2009.

[16] Voigt, Konrad, "Structural Graph-based Metamodel Matching," Ph.D. dissertation, Dresden University, 2011.

[17] J.-R. Falleri, M. Huchard, M. Lafourcade, and C. Nebut, "Metamodel matching for automatic model transformation generation," in *MODELS*, 2008, pp. 326–340.

[18] S. Melnik, H. Garcia-Molina, and E. Rahm, "Similarity flooding: A versatile graph matching algorithm and its application to schema matching," in *ICDE*, 2002, pp. 117–128.

[19] K. Voigt and T. Heinze, "Metamodel matching based on planar graph edit distance," in *Theory and Practice of Model Transformations*, 2010, pp. 245–259.

[20] J. Cadavid, B. Baudry, and H. Sahraoui, "Searching the Boundaries of a Modeling Space to Test Metamodels," in *IEEE ICST*, 2012, pp. 131–140.

[21] E. Batot and H. Sahraoui, "A Generic Framework for Model-set Selection for the Unification of Testing and Learning MDE Tasks," in *MODELS*, 2016, pp. 374–384.

[22] S. Sen, B. Baudry, and J.-M. Mottu, "On Combining Multi-formalism Knowledge to Select Models for Model Transformation Testing," in *ICST, IEEE International Conference on Software Testing, Verification and Validation*, 2008, pp. 328–337.

[23] B. Henderson-Sellers, *Object-Oriented Metrics: Measures of Complexity*. Prentice Hall, 1996.

[24] E. Van Emden and L. Moonen, "Java quality assurance by detecting code smells," in *Conference on Reverse Engineering*. IEEE, 2002, pp. 97–106.

[25] M. Mitzenmacher and E. Upfal, *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. Cambridge University Press, 2005.

[26] "Qualitas corpus," qualitascorpus.com/docs/catalogue/20130901/index.html, accessed: 2018-05-29.

[27] "Metrics tool," metrics.sourceforge.net, accessed: 2018-05-29.

[28] "R software," www.r-project.org, accessed: 2018-05-29.

[29] M. Weller, A. Chateau, and R. Giroudeau, "Exact approaches for scaffolding," *BMC Bioinformatics*, vol. 16, no. 14, pp. 1471–2105, 2015.

[30] C. Brun and A. Pierantonio, "Model differences in the eclipse modeling framework," *UPGRADE, The European Journal for the Informatics Professional*, vol. 9, no. 2, pp. 29–34, 2008.

[31] G. Szárnyas, Z. Kovári, Á. Salánki, and D. Varró, "Towards the characterization of realistic models: evaluation of multidisciplinary graph metrics," in *MODELS*, 2016, pp. 87–94.

[32] R. W. Hamming, "Error detecting and error correcting codes," *Bell System technical journal*, vol. 29, no. 2, pp. 147–160, 1950.

[33] A. Singhal, "Modern information retrieval: A brief overview," *IEEE Data Engineering Bulletin*, vol. 24, no. 4, pp. 35–43, 2001.

[34] V. Levenshtein, "Binary codes capable of correcting deletions, insertions, and reversals," in *Soviet physics doklady*, vol. 10, no. 8, 1966, pp. 707–710.

[35] G. Kishi, "On centrality functions of a graph," in *Graph Theory and Algorithms*. Springer, 1981, pp. 45–52.

[36] L. Page, S. Brin, R. Motwani, and T. Winograd, "The PageRank citation ranking: bringing order to the web," Stanford InfoLab, Tech. Rep., 1999.

[37] R. Kollmann and M. Gogolla, "Metric-based selective representation of uml diagrams," in *CSMR*. IEEE, 2002, pp. 89–98.

[38] R. von Mises and H. Pollaczek-Geiringer, "Praktische verfahren der gleichungsauflösung." *ZAMM-Journal of Applied Mathematics and Mechanics*, vol. 9, no. 2, pp. 152–164, 1929.

[39] M. Roy, S. Schmid, and G. Trédan, "Modeling and measuring Graph Similarity: the Case for Centrality Distance," in *FOMC*, 2014, pp. 47–52.

[40] F. Murtagh, "Multidimensional clustering algorithms," *Compstat Lectures, Vienna: Physika Verlag, 1985*, 1985.

[41] R Development Core Team, *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, 2008.

[42] F. Aurenhammer, "Voronoi diagrams a survey of a fundamental geometric data structure," *CSUR, ACM Computing Surveys*, vol. 23, no. 3, pp. 345–405, 1991.

[43] "Download comodi," github.com/ferdjoukh/comodi, accessed: 2018-05-29.

[44] C. Dallard, M. Weller, A. Chateau, and R. Giroudeau, "Instance guaranteed ratio on greedy heuristic for genome scaffolding," in *COCOA, Combinatorial Optimization and Applications*, 2016, pp. 294–308.

[45] J.-M. Mottu, S. Sen, J. Cadavid, and B. Baudry, "Discovering model transformation pre-conditions using automatically generated test models," in *ISSRE, International Symposium on Software Reliability Engineering*, 2015, pp. 88–99.

An Auto-active Approach to Develop Correct Logic-based Graph Transformations

Amani Makhlouf, Christian Percebois, Hanh Nhi Tran

University of Toulouse
IRIT Laboratory
Toulouse, France
Email: {Amani.Makhlouf | Christian.Percebois | Hanh-Nhi.Tran}@irit.fr

***Abstract*—We aim at assisting developers to write, in a Hoare style, provably correct graph transformations expressed in the $\mathcal{ALCQ}$ Description Logic. Given a postcondition and a transformation rule, we compute the weakest precondition for developers. However, the size and quality of this formula may be complex and hard to grasp. We seek to reduce the weakest precondition's complexness by a static analysis based on an alias calculus. The refined precondition is presented to the developer in terms of alternative formulae, each one specifying a potential matching of the source graph. By choosing some alternatives that correspond to his intention, the developer can interact with an auto-active program verifier, which continuously ensures the correctness of the resulting Hoare triple.**

***Keywords*–Graph transformation; weakest precondition calculus; static analysis; alias calculus; auto-active program verifier.**

I. Introduction

All approaches applying production rules to a graph require to implement a binary relation between a source graph and a target graph. In the theory of algebraic graph transformations, Habel and Pennemann [2] defined nested graph conditions as a graphical and logical formalism to specify graph constraints by explicitly making use of graphs and graph morphisms. Nested conditions have the same expressive power as Courcelle's first-order graph logic [2][3][4]. However, they need to be derived into specific inference rules in order to be proved in a specific theorem-prover that suits them [5][6]. Moreover, this transformation requires the proof of a sound and complete proof system for reasoning in the proposed logic.

The proof of the completeness part of Pennemann's transformation for nested conditions was done by Lambers and Orejas [7] thanks to a tableau reasoning method. The authors introduce nested tableaux, an extension of usual tableaux, to take into account nested conditions. Their proof system requires an $\mathcal{M}$-adhesive category allowing some compatibility of pushouts and pullbacks along $\mathcal{M}$-morphisms. Recently, $\mathcal{M}$-adhesive transformation systems have been generalized to $\mathcal{M}, \mathcal{N}$-adhesive ones [8] to cover graph programs dealing with node relabeling as done in GP [9].

Another way to express and reason about graph properties is to directly encode graphs in terms of some existing logic [10]. This solution leads to consider connections between graph constraints and first-order graph formulae. Adopting this approach, we define graphs axiomatically by $\mathcal{ALCQ}$ Description Logic (DL) predicates [11] and manipulate them with specific statements. In this way, we designed a non-standard imperative programming language named Small-t$\mathcal{ALC}$ dedicated to transform labeled directed graphs. The suffix is limited to $\mathcal{ALC}$ because this logic is prototypical of DLs.

Despite the above differences from algebraic graph transformations, we point out the common idea to use satisfiability solvers to prove rules'correctness. This technique requires to assign a predicate transformer to a rule in order to compute the rule's weakest precondition. The setup is rather traditional: given a Hoare triple $\{P\}S\{Q\}$, we compute the weakest (liberal) precondition $wp(S, Q)$ of the rule transformation statements S with respect to the postcondition Q, and then verify the implication $P \Rightarrow wp(S, Q)$. The correctness of the rule is proved by a dedicated tableau reasoning, which is sound, complete and which results in a counter-example when a failure occurs. This verification can be realized in an auto-active mode [12] where developers annotate their code with specifications to facilitate the automatic verification of the program's correctness. Some verifiers like AutoProof [13] or Dafny [14] use this approach for object-oriented languages.

Aiming to assist developers in writing provably correct transformations [15], we adopt this auto-active approach to provide more interactions with developers to produce a Hoare triple, and thus benefit from their guidance and give them more feedback. In this context, we propose to statically calculate the weakest precondition based on an alias calculus in order to suggest precondition formulae that are easier to understand but still ensuring the correctness of the Hoare-triple. The result is presented to developers in a disjunctive normal form. Each conjunction of positive and negative literals specifies a potential matching of the source graph. By letting developers interactively choose a conjunction as a premise that reflects the rule's intention, our approach can filter and reduce some combinatorial issues.

This paper presents an extension of our work originally reported in Proceedings of the Twelfth International Conference on Software Engineering Advances [1]. Section II first defines logic-based formulae to annotate pre- and postconditions of a transformation rule. This choice yields manageable proof obligations in a Hoare's style for rules'correctness. Then, we introduce in Section III Small-t$\mathcal{ALC}$ atomic statements that manipulate graph structures. Each statement is characterized by a weakest precondition with respect to a given postcondition. On the basis of an alias calculus that is presented in Section IV-A, we show in Section IV-B how to reduce some combinatorial issues while ensuring the program correctness by finely analyzing the weakest precondition. This leads to an auto-active verification of a Hoare triple, which is sketched out in Section V. In Section VI we present our integrated development environment consisting of various tools to assist developers in writing, executing, testing and reasoning about graph transformations. We finally give some discussions on related work in Section VII and wrap up the paper with a conclusion and possible improvements in Section VIII.

II. Logic-based conditions

Slightly diverged from the standard approach, we choose a set-theoretic approach for our transformation system [16]. The basic idea is to specify sets of nodes and edges of a subgraph using a fragment of first-order logic. It turns out that replacing graph patterns by graph formulae yields manageable proof obligations for rules'correctness in a Hoare style $\{P\}S\{Q\}$ [10]. A precondition formula P designates a subgraph matching a substructure that should exist in the source graph. The postcondition Q requires the existence of the subgraph represented by Q in the target graph. For instance, consider a rule requiring that: (1) x must be a node (individual) not connected by the relation (role) R to a node y; (2) y is of class (concept) C; (3) x is linked to at most three successors (qualified number of restrictions) of class C via R. This precondition can be expressed by the logic formula $x \neg R\, y \ \wedge\ y : C \wedge\ x : (\leq\ 3\ R\ C)$.

At this point, readers familiar with Description Logics (DLs) may recognize a DL formula. Labeled directed graphs can be directly modeled by entities of DLs, a family of logics for modeling and reasoning about relationships in a domain of interest [17]. Most DLs are decidable fragments of first-order logic. They are organized around three kinds of entities: individuals, roles and concepts. Individuals are constants in the domain, roles are binary relations between individuals and concepts are sets of individuals. Applied to our graphs, individuals are nodes labeled with concepts and roles are edges. Accordingly, pre- and post-assertions are interpreted as graphs by using unary predicates for nodes and binary predicates for edges. The correctness of a graph transformation rule is checked by assigning to each of its statements a predicate transformer in order to compute the corresponding weakest precondition.

To design our own experimental graph transformation language, we chose the $\mathcal{ALCQ}$ logic, an extension of the standard DL Attributive Language with Complements ($\mathcal{ALC}$) [18], which allows qualifying number restrictions on concepts ($\mathcal{Q}$). $\mathcal{ALCQ}$ is based on a three-tier framework: concepts, facts and formulae. The concept level enables to determine classes of individuals ($\emptyset, C, \neg C, C1 \cup C2$ and $C1 \cap C2$). The fact level makes assertions about individuals owned by a concept ($i : C, i : \neg C, i : (\leq\ n\ R\ C)$ and $i : (\geq\ n\ R\ C)$), or involved in a role ($i\ R\ j$ and $i\ \neg R\ j$). The third level is about formulae defined by a Boolean combination of $\mathcal{ALCQ}$ facts ($f, \neg f, f1 \wedge f2$ and $f1 \vee f2$).

Figure 1 depicts a model (graph) satisfying the previous precondition $x \neg R\, y \ \wedge\ y : C \wedge\ x : (\leq\ 3\ R\ C)$. In this graph, the white circles designate the nodes variables x and y manipulated by the formula. Nodes variables refer (by a dotted edge) to real nodes represented by black circles. The «◉» node outlines a concept labeled with C. Note that the subgraph having two anonymous nodes each one outfitted with an incoming edge from x and an outgoing edge to the concept C is a model, which checks the fact $x : (\leq\ 3\ R\ C)$.

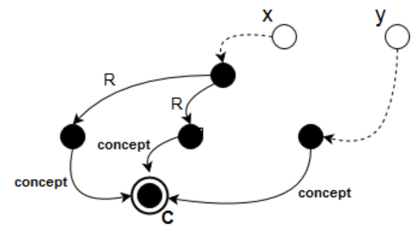


Figure 1. Model satisfying the precondition $x \neg R\, y \ \wedge\ y : C \wedge\ x : (\leq\ 3\ R\ C)$

Our formulae contain free variables that assign references to nodes in a graph. Equality and inequality assertions can be used to define constraints on the value of these variables. If x and y are node variables, $x = y$ means that x and y refer to the same node and $x \neq y$ means that x and y are distinct. The inequality relationship enforces injective graph morphisms.

III. The Small-t$\mathcal{ALC}$ language

The $\mathcal{ALCQ}$ formulae presented in the previous section have been plugged into our Small-t$\mathcal{ALC}$ imperative language and used in atomic transformation actions on nodes (individuals) and edges (roles), as well as in traditional control-flow constructs as loops ($while$) and conditions ($if...then...else...$). In the transformation code, statements manipulate node variables, which are bound to the host graph's nodes during the transformation's execution.

We have defined five atomic Small-t$\mathcal{ALC}$ statements according to the following grammar where i and j are node variables, C is a concept name, R is a role name, F is an $\mathcal{ALCQ}$ formula and v is a list of node variables:

atomic_statement ::=

$add(i : C)$	(node labeling)
$\mid delete(i : C)$	(node unlabeling)
$\mid add(i\ R\ j)$	(edge labeling)
$\mid delete(i\ R\ j)$	(edge unlabeling)
$\mid select\ v\ with\ F$	(assignment)

The first four statements modify the graph structure by changing the labeling of nodes and edges. Note that since we consider a set-theoretic approach, the statements $add(i : C)$ and $add(i\ R\ j)$ have no effects if i belongs to the set C and (i, j) to R respectively. Hence, no parallel edges with the same label are allowed. An original construct is the $select$ statement that non-deterministically binds node variables to nodes in the subgraph that satisfies a logic formula. This assignment is used to handle the selection of specific nodes where the transformations are requested to occur. For instance, $select\ i\ with\ i : C$ selects a node labeled with C. If the selection is satisfied the execution continues normally with the value of the node variable i. Otherwise, the execution meets an error situation.

A Small-t$\mathcal{ALC}$ program is organized into rules. A rule is structured into three parts: a precondition, the transformation code and a postcondition. Small-t$\mathcal{ALC}$ uses the classical control structures to enable sequential composition, branching and iteration of the atomic statements. Inside a rule, the developer can write an ordered list of statements $s1; s2$; use the statements if or $if - else$ to express a choice between alternatives or the statement $while$ for repeating actions. We illustrate in Figure 3 an example of a transformation rule written in Small-t$\mathcal{ALC}$. The rule r first selects a node n of concept A that is R-linked to a. Then, it deletes this link and removes a from the concept A.

```
rule r {
  pre: (a : A) ∧ a : (≥ 3 R A);
  select n with (a R n) ∧ (n : A);
  delete(a R n);
  delete(a : A);
  post: (a : ¬A) ∧ a : (≥ 2 R A);
}
```

Figure 3. Example of a Small-t$\mathcal{ALC}$ rule

The inference rules in Figure 2 define the axiomatic semantics of Small-t$\mathcal{ALC}$. Each rule consists of a premise and a conclusion separated by a horizontal bar. Some ones display prominently the substitutions in the form $P[E\backslash V]$ to indicate that V is replaced by E in P when updating nodes and edges of a graph. For instance, the inference rule ADDC can be interpreted that if the assertion P is valid when substituting the concept $C+i$ for the concept C then P is valid after executing the statement $add(i : C)$. We can say that ADDC is similar to the rule defining the semantics of the assignment $V := E$ in traditional imperative programs. The same observation can be done for DELC defining the semantics of the statement $delete(i : C)$, which modifies the interpretation for C in order to delete the node denoted by i. Rules ADDR and DELR define the semantics of $add(i\ R\ j)$ and $delete(i\ R\ j)$ to add and delete respectively a pair of nodes (i, j) to/from the set of pairs connected by R.

The rules SEQ, IF, IF-ELSE and WHILE give the semantics for the Small-t$\mathcal{ALC}$ control structures. Conditionals in Small-t$\mathcal{ALC}$ are specific DL predicates, which can be considered as Booleans queries on a knowledge base of concept assertions and role assertions. Evaluating a conditional Boolean expression is without side-effects. The statement $select\ v\ with\ F$ is more specific. As we seek and assign individuals when checking the formula F, the inference rule SELECT specifies that any choice for a list v of nodes that satisfies the condition F must provide the postcondition Q assuming the precondition P. If no instance satisfies F, the semantics is blocked.

To shape a sequencing of transformation steps, Small-t$\mathcal{ALC}$ rules will be called inside the $main$ function, which is the entry-point of a Small-t$\mathcal{ALC}$ program. Two separate axiomatic definitions CALL and CALL! are defined for rule invocations inside the $main$ function. Given a rule r and a statement S, we denote $body(r) = S$ in order to ascertain the logical relation of assertions around the execution of the body of r. Note that in Small-t$\mathcal{ALC}$ we only consider rules without parameters. The inference rule CALL refers to a simple rule invocation and says that if we can show that the relation $\{P\}S\{Q\}$ is true where $S = body(r)$, then the relation $\{P\}r\{Q\}$ is true. Thanks to the inference rule CALL!, a body of a rule can be executed many times. Such a call corresponds to an iteration as long as a subgraph matches the rule's precondition formula. To express the semantics of CALL!, we conclude that the current rule call is correct (i.e., $\{P\}r\{Q\}$) if we assume that the previous calls are correct as well (i.e., $\{P\}r\{Q\} \vdash \{P\}S\{Q\}$, which can be interpreted as followed: the sentence $\{P\}S\{Q\}$ is a syntactic consequence ($\vdash$) of the assumption $\{P\}r\{Q\}$).

We aim at using a Hoare-like calculus to prove that Small-t$\mathcal{ALC}$ graph programs are correct. This verification process is based on a weakest (liberal) precondition (wp) calculus [19]. Each Small-t$\mathcal{ALC}$ statement S is assigned to a predicate transformer yielding an $\mathcal{ALCQ}$ formula $wp(S, Q)$ assuming

$$\frac{}{\{P[C+i \setminus C]\}\ \texttt{add}\ (i:C)\ \{P\}}\ (\textsc{add}C) \qquad \frac{}{\{P[C-i \setminus C]\}\ \texttt{delete}\ (i:C)\ \{P\}}\ (\textsc{del}C)$$

$$\frac{}{\{P[R+(i,j) \setminus R]\}\ \texttt{add}\ (i\ R\ j)\ \{P\}}\ (\textsc{add}R) \qquad \frac{}{\{P[R-(i,j) \setminus R]\}\ \texttt{delete}\ (i\ R\ j)\ \{P\}}\ (\textsc{del}R)$$

$$\frac{P\ \wedge \forall v(F \Rightarrow Q)}{\{P\}\ \texttt{select}\ v\ \texttt{with}\ F\ \ \{Q\}}\ (\textsc{select}) \qquad \frac{\{P\}\ s1\ \{Q\} \quad \{Q\}\ s2\ \{R\}}{\{P\}\ s1; s2\ \{R\}}\ (\textsc{seq})$$

$$\frac{\{P \wedge c\}\ s\ \{Q\} \quad \{P \wedge \neg c\} \Rightarrow \{Q\}}{\{P\}\ \texttt{if}\ c\ \texttt{then}\ s\ \{Q\}}\ (\textsc{if}) \qquad \frac{\{P \wedge c\}\ s1\ \{Q\} \quad \{P \wedge \neg c\}\ s2\ \{Q\}}{\{P\}\ \texttt{if}\ c\ \texttt{then}\ s1\ \texttt{else}\ s2\ \{Q\}}\ (\textsc{if-else})$$

$$\frac{\{P \wedge c\}\ s\ \{P\}}{\{P\}\ \texttt{while}\ c\ \texttt{do}\ s\ \{P \wedge \neg c\}}\ (\textsc{while})$$

$$\frac{\{P\}\ S\ \{Q\} \quad body(r) = S}{\{P\}\ r\ \{Q\}}\ (\textsc{call}) \qquad \frac{\{P\}\ r\ \{Q\} \vdash \{P\}\ S\ \{Q\} \quad body(r) = S}{\{P\}\ r\ \{Q\}}\ (\textsc{call!})$$

Figure 2. Axiomatic semantics of Small-t$\mathcal{ALC}$

the postcondition Q. The correctness of a program prg with respect to Q is established by proving that the given precondition P implies the weakest precondition: every model that satisfies P also satisfies $wp(prg, Q)$. Weakest preconditions of Small-t$\mathcal{ALC}$ statements are given in Figure 4.

$$
\begin{aligned}
& wp(add\ (i : C), Q) = Q[C + i \backslash C] \\
& wp(delete\ (i : C), Q) = Q[C - i \backslash C] \\
& wp(add\ (i\ R\ j), Q) = Q[R + (i, j) \backslash R] \\
& wp(delete\ (i\ R\ j), Q) = Q[R - (i, j) \backslash R] \\
& wp(select\ v\ with\ F, Q) = \forall v(F \Rightarrow Q) \\
& wp(s1; s2,\ Q) = wp(s1, wp(s2,\ Q)) \\
& wp(if\ c\ then\ s1, Q) = (c \Rightarrow wp(s1, Q)) \wedge (\neg c \Rightarrow Q) \\
& wp(if\ c\ then\ s1\ else\ s2, Q) = (c \Rightarrow wp(s1, Q)) \\
& \qquad\qquad\qquad\qquad \wedge (\neg c \Rightarrow wp(s2, Q)) \\
& wp(\{inv\}\ while\ c\ do\ s, Q) = inv
\end{aligned}
$$

Figure 4. Small-t$\mathcal{ALC}$ weakest preconditions

Small-t$\mathcal{ALC}$ axiomatic semantics for add and $delete$ statements introduces substitutions, which build formulae that no longer belong to the $\mathcal{ALCQ}$ logic: $C + i$ in ADDC, $C - i$ in DELC, $R + (i, j)$ in ADDR and $R - (i, j)$ in DELR do not represent concepts and roles anymore. This means that the weakest precondition calculus computes predicates, which are not closed under substitutions with respect to $\mathcal{ALCQ}$ [20]. To resolve this situation, substitutions are considered as constructors for concepts and roles and should be eliminated by predicate transformers. For instance:

$$
\begin{aligned}
wp(add(i : C), x : C) &= x : C[C + i \backslash C] \\
&= x : (C + i) \\
&= x : C \vee x = i.
\end{aligned}
$$

IV. SPECIFICATION EXTRACTION

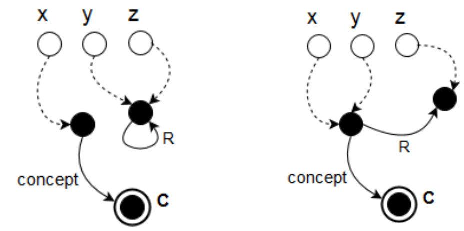
The conventional precondition calculus presented in the previous section does not take into account particular situations of a transformation program and thus may result in a complex precondition. In this section, we look at how the precondition's formula can be improved to be more specific and simple on the basis of alias calculus.

A. Alias calculus

The principle of alias calculus was proposed by Bertrand Meyer in order to decide whether two reference expressions appearing in a program might, during some execution, have the same value, meaning that the associated references are attached to the same object [21].

Since our rewriting system allows non-injective morphisms, two or more node variables may reference to the same node in a graph. On the other hand, a node variable can be assigned to a random node of the graph. This is one reason why a Small-t$\mathcal{ALC}$ formula can be represented by several graph patterns. For example, Figure 5 shows two potential models satisfying the formula $x : C\ \wedge\ y\ R\ z$. In Figure 5a, y and z refer to the same node. In 5b, y and z are different but x and y are combined.

In this regard, for a transformation program, we apply an alias calculus to determine the node variables that can never refer to the same node. Discerning such specific circumstances helps to discard later unsatisfied subformulae of the weakest precondition. Thus, our method consists in assigning to each node variable x, a set of other node variables that may reference



(a) $y = z, x \neq y$ (b) $y \neq z, x = y$

Figure 5. Example of models satisfying the formula $x : C\ \wedge\ y\ R\ z$

to the same node in the graph as x. We identify four atomic conditions in which two individuals x and y can never refer to the same node in the graph:

- $x \neq y$
- $\exists C\ /x : C \wedge y : \neg C$
- $\exists R.\exists z\ /x\ R\ z \wedge y\ \neg R\ z$
- $\exists R.\exists z\ /z\ R\ x \wedge z\ \neg R\ y$

The first case ($x \neq y$) states that x and y are naturally distinct so they can never be assigned to the same node. The second one asserts that x and y belong to two complement subsets C and $\neg C$. The same applies to the last two cases where the nodes connected by R and $\neg R$ refer to two disjoint subsets R and $\neg R$.

For each of the above four conditions, x and y are said to be *non-possibly equivalent* nodes. We note this relation by $x \not\simeq y$. As a result we assert that $x \not\simeq y \Rightarrow x \neq y$. However, no conclusion can be drawn from the *possibly equivalent* relation $x \simeq y$.

Consider, as a simple example, the following formula that is presented in the disjunctive normal form: $(x = y \wedge x\ R\ y) \vee (x : C \wedge x\ \neg R\ y)$, and suppose that a static analysis deduces from the code that x and y are non-possibly equivalent, which means that $x \neq y$. As a result, the initial formula can be reduced to $x : C \wedge x\ \neg R\ y$ because the first conjunction $x = y\ \ \wedge x\ \ R\ \ y$ can never be true in this case. In the section that follows, we show how this calculus helps in reducing the complexness of the weakest precondition.

B. Precondition extraction

To formally verify the correctness of a Small-t$\mathcal{ALC}$ graph transformation, besides the code, the program's pre- and postconditions must be properly specified. This task may not be easy for novice developers, so a suggestion of a valid precondition that corresponds to a given code and a postcondition would be useful to them.

Since the computed weakest precondition is often very complex and hard to comprehend, we propose a finer static analysis on the basis of the alias calculus of the program to achieve a simpler precondition. The resulting precondition P is presented in a disjunctive normal form (DNF) where each conjunction of P can be considered as a valid precondition on its own. The analysis consists first in converting the postcondition Q to DNF, i.e., $Q = \vee Q_i$ where $Q_i = \wedge q_j$ is a conjunction of facts, then calculating for each statement and for each conjunction Q_i the weakest precondition. This process maintains correctness because $wp(S, Q_1)\ \vee\ wp(S, Q_2)\ \ \Rightarrow wp(S, Q_1 \vee Q_2)$. In each and every step, the formula of the

$wp(S, Q_i)$ may be filtered by discarding subformulae according to the identified non-possibly equivalent node variables. A precondition P is obtained such that $P \Rightarrow wp(prg, Q)$, which makes the transformation program prg correct. This process is applied to *add* and *delete* statements as detailed in Section IV-B1. Regarding the *select* statement, *wp* is reduced differently as presented later in Section IV-B2.

1) The add *and* delete *statements:*
Let us consider first the *add(i : C)* statement. Its weakest precondition with respect to the postcondition $x : C$ is $x : C \vee x = i$, which means that either the node x was already of concept C before adding i to C, or x and i are equal. Knowing that x and i are non-possibly equivalent, it can be stated that $x \neq i$, and so the weakest precondition can be reduced to the first subformula *x : C* of the disjunction.

A more glaring example is reducing the weakest precondition of the *add(i R j)* statement with respect to the postcondition $Q = x : (\leq \ n \ R \ C)$, which indicates that there are at most n edges labeled R outgoing from the node x to nodes of concept C. Adding an R-edge between i and j may have a direct impact on Q regarding the concept of j, the existence of a relation between i and j and the equality between i and x.
Hence, $wp(add(i \ R \ j), x : (\leq \ n \ R \ C)) =$

$$
\begin{aligned}
&(x = i \ \wedge \ j : C \ \wedge \ i \neg R \ j \ \wedge \ x : (\leq \ (n-1) \ R \ C)) \\
&\vee \ (x \neq i \ \wedge \ x : (\leq \ n \ R \ C)) \\
&\vee \ (j : \neg C \ \wedge \ x : (\leq \ n \ R \ C)) \\
&\vee \ (i \ R \ j \ \wedge \ x : (\leq \ n \ R \ C)) \\
&\vee \ (x : (\leq \ (n-1) \ R \ C))
\end{aligned}
$$

Knowing that $x \not\simeq i$ or $j : \neg C$, the first conjunction $x = i \wedge j : C \wedge i \neg R \ j \wedge x : (\leq \ (n-1) \ R \ C)$ can be discarded as it will never be satisfied in this case. Furthermore, the whole formula of the wp can be reduced to $x : (\leq \ n \ R \ C)$ according to the second and third conjunctions, which indicate that the number of restrictions remains unchanged in case one of these two conditions is satisfied.

We illustrated how to reduce the *wp* with respect to a postcondition composed of a single fact. In case of a postcondition consisting of a conjunction of facts, only the facts that manipulate the same concepts and roles given in the statement parameters are identified as a first step. For example, adding an instance to a concept ($add(i : \underline{C})$) results in considering in the given postcondition only the facts that manipulate this concept ($x : \underline{C}, x : \ \neg\underline{C}, x : (\leq \ n \ R \ \underline{C})$).

Tables I and III represent the preconditions calculated by our static analyzer for the statement $add(i \ : \ C)$ and $add(i \ R \ j)$ respectively. For each statement s, we show in the second column the facts that should be identified in the postcondition to derive a precondition. The third column shows the standard weakest precondition $wp(s, f)$ of the statement s with respect to an identified fact f. To simplify this formula, we present in the fourth column the conditions that allow to discard some conjunctive clauses of the wp. The resulting formula is presented in the last column.

Consider the first row of the Table III. If a fact $x \ R \ y$ is identified within the postcondition during calculation, we look for simplifying $wp(add(i \ R \ j), x \ R \ y) = (x = i \wedge y = j) \vee x \ R \ y$. If the alias calculus asserts that at least one of the conditions $x \not\simeq i$ or $y \not\simeq j$ is true, wp is reduced to $x \ R \ y$.

As observed in Tables I and III, many complex disjunctions in the *wp* can be reduced to only one conjunction on the basis of a condition calculated by the alias calculus or a condition given explicitly in the postcondition. Note that the results of the *delete(i : C)* and *delete(i R j)* statements are similar to the *add* statements and are respectively described in Tables II and IV.

2) The select *statement:*
So far, the static analysis transforms the predicate Q into a new predicate P regarding statements already presented. However, it operates differently when it comes to the *select* statement where $wp(select \ v \ with \ F, Q) = \forall v \ (F \Rightarrow Q)$. The weakest precondition here involves two formulae that may be complex: F given by the *select*, and the postcondition Q. Consequently, the implication $F \Rightarrow Q$ makes the wp more obscure for the developer. In this case, the static analyzer simplifies the wp by eliminating this implication as further detailed below.

For each conjunction Qi of the postcondition Q, the static analysis isolates first the facts that manipulate the node variables v of the *select* statement. Let Qi_v be the conjunctive formula of these identified facts, and $Qi_{v'}$ the conjunctive formula of the others facts, so that $Qi = Qi_v \wedge Qi_{v'}$. For example, given a formula $Q_1 = x \ R \ y \wedge y : C$ and the statement $select \ x \ with \ x : C$, we have $Q_{1v} = x \ R \ y$ and $Q_{1v'} = y : C$.

Then, the static analysis checks, via our logic formula evaluator, if the implication $\forall v \ (F \Rightarrow Qi_v)$ holds. If so, the precondition $wp(select \ v \ with \ F, Qi) = \forall v \ (F \Rightarrow Qi)$ is reduced to Qi without affecting the validity of the Hoare triple as $Qi \Rightarrow wp(select \ v \ with \ F, Qi)$. Conversely, the non-validity of the implication $\forall v \ (F \Rightarrow Qi_v)$ results in transforming Qi to the predicate $false$ ($\perp$) so that nothing can be concluded about the transformation correctness. This situation is meant to warn the developer that there are inconsistencies in his transformation between the $select$ statement and the predicate formula Q. These cases are given in Table V for a conjunctive formula Qi.

To clarify the idea, consider an example of a code consisting of the statement $select \ i \ with \ i : C$. First, suppose that the given postcondition is $Q = (i : C) \wedge (j : C)$. So we have $F = i : C$ and $Q_v = i : C$ as $i : C$ is the only fact that manipulates the selected node variable i in the postcondition. The implication $i : C \Rightarrow i : C$ is obviously true, so the precondition is reduced to Q.
Now consider another postcondition for the same code: $Q = (i \ R \ j) \wedge (j \ : \ C)$. In this case, the implication between $F = i : C$ and $Q_v = i \ R \ j$ does not hold. Hence, the static analyzer returns $false$ as a precondition, and so this transformation can not be executed because no state can satisfy the precondition as mentioned above.

3) The other statements:
We presented how the static analyzer filters the weakest precondition of an atomic statement with respect to each conjunction $Q_i = \wedge q_j$ of Q where $Q = \vee Q_i$. The precondition P of a sequence of statements $s1; s2$ is computed conventionally as it is presented above ($wp(s1, wp(s2, Q))$). Similarly, the extracted precondition of the $if \ c \ then \ s1 \ else \ s2$ statement is its weakest precondition transformed into a DNF formula: $(c \wedge wp(s1, Q)) \vee \ (\neg c \wedge wp(s2, Q))$.

Apart from loops, weakest preconditions can be computed automatically as it is presented in this section. How-

TABLE I. WEAKEST PRECONDITION'S FILTERING FOR THE $add(i : C)$ STATEMENT

Statement	Identified fact	wp	Condition	Precondition
$add(i : C)$	$x : C$	$x : C \vee x = i$	$x \not\simeq i$	$x : C$
	$x : \neg C$	$x : \neg C \wedge x \neq i$	$x \not\simeq i$	$x : \neg C$
	$x : (\leq n\ R\ C)$	$(x\ R\ i \wedge i : \neg C \wedge x : (\leq (n-1)\ R\ C))$ $\vee (x \neg R\ i \wedge x : (\leq n\ R\ C))$ $\vee (i : C \wedge x : (\leq n\ R\ C))$ $\vee (x : (\leq (n-1)\ R\ C))$	$x \neg R\ i$	$x : (\leq n\ R\ C)$

TABLE II. WEAKEST PRECONDITION'S FILTERING FOR THE $delete(i : C)$ STATEMENT

Statement	Identified fact	wp	Condition	Precondition
$delete(i : C)$	$x : \neg C$	$x : \neg C \vee x = i$	$x \not\simeq i$	$x : \neg C$
	$x : C$	$x : C \wedge x \neq i$	$x \not\simeq i$	$x : \neg C$
	$x : (\geq n\ r\ C)$	$(x\ r\ i \wedge i : C \wedge x : (\geq n+1\ r\ C))$ $\vee (x \neg r\ i \wedge x : (\geq n\ r\ C))$ $\vee (i : \neg C \wedge x : (\geq n\ r\ C))$ $\vee (x : (\geq n+1\ r\ C))$	$x \neg r\ y$	$x : (\geq n\ r\ C)$

TABLE III. WEAKEST PRECONDITION'S FILTERING FOR THE $add(i\ R\ j)$ STATEMENT

Statement	Identified fact	wp	Condition	Precondition
$add(i\ R\ j)$	$x\ R\ y$	$(x = i \wedge y = j) \vee x\ R\ y$	$x \not\simeq i \vee y \not\simeq j$	$x\ R\ y$
	$x \neg R\ y$	$(x \neq i \vee y \neq j) \wedge (x \neg R\ y)$	$x \not\simeq i \vee y \not\simeq j$	$x \neg R\ y$
	$x : (\leq n\ R\ C)$	$(x = i \wedge j : C \wedge i \neg R\ j \wedge x : (\leq (n-1)\ R\ C))$ $\vee (x \neq i \wedge x : (\leq n\ R\ C))$ $\vee (j : \neg C \wedge x : (\leq n\ R\ C))$ $\vee (i\ R\ j \wedge x : (\leq n\ R\ C))$ $\vee (x : (\leq (n-1)\ R\ C))$	$x \not\simeq i \vee j : \neg C$	$x : (\leq n\ R\ C)$

TABLE IV. WEAKEST PRECONDITION'S FILTERING FOR THE $delete(i\ R\ j)$ STATEMENT

Statement	Identified fact	wp	Condition	Precondition
$delete(i\ R\ j)$	$x \neg R\ y$	$(x = i \wedge y = j) \vee x \neg R\ y$	$x \not\simeq i \vee y \not\simeq j$	$x \neg R\ y$
	$x\ R\ y$	$(x \neq i \vee y \neq j) \wedge (x\ R\ y)$	$x \not\simeq i \vee y \not\simeq j$	$x\ R\ y$
	$x : (\geq n\ R\ C)$	$(x = i \wedge j : C \wedge i\ R\ j \wedge x : (\geq n+1\ R\ C))$ $\vee (x \neq i \wedge x : (\geq n\ R\ C))$ $\vee (j : \neg C \wedge x : (\geq n\ R\ C))$ $\vee (i \neg R\ j \wedge x : (\geq n\ R\ C))$ $\vee (x : (\geq n+1\ R\ C))$	$x \not\simeq i \vee j : \neg C$	$x : (\geq n\ R\ C)$

TABLE V. REDUCING THE WP OF THE $select$ STATEMENT

Statement	Postcondition	wp	Condition	Precondition
$select\ v\ with\ F$	Qi	$\forall v\ (F \Rightarrow Qi)$	$\forall v\ (F \Rightarrow Qi_v)$	Qi
			$\forall v\ (F \not\Rightarrow Qi_v)$	$\perp$

ever, it is more complicated when it comes to the $while$ statement. In fact, $while\ c\ do\ s$ is semantically equivalent to $if\ c\ then\ \{s; while\ c\ do\ s\}\ else\ skip$, then its weakest precondition is a recursive equation of the form: $wp(while\ c\ do\ s, Q) = (c \Rightarrow wp(s, wp(while\ c\ do\ s, Q))) \vee (\neg c \Rightarrow Q)$. For this reason, the weakest precondition is approximated by a verification condition, which considers the invariant of the $while$ statement as it is shown in the

TABLE VI. VERIFICATION CONDITION OF THE *while* STATEMENT

Statement	Postcondition	Verification condition	Precondition
$\{inv\}\ while\ c\ do\ s$	Q	$(inv \wedge \neg c \Rightarrow Q)$ $\wedge\ (inv \wedge c \Rightarrow wp(s, inv))$ $\wedge\ inv$	$DNF(inv)$
		else	$\bot$

third column of the Table VI. Basing on the given invariant inv, our logic formula evaluator checks whether all of the implications presented in the third column hold. In this case, the precondition of the loop is simplified by inv. Otherwise, Q is transformed into the predicate $false$ ($\bot$) as the given invariant does not satisfy the loop verification condition.

The final result of the precondition will be presented as a DNF formula that expresses different possible alternatives. Each alternative represents a conjunction of facts, constituting a graph that matches a subgraph of the source graph on which the transformation rule is applied.

We filter the weakest precondition by discarding conjunctive clauses that are invalid. This reduction leads to a precondition P stronger than the weakest precondition $wp(S, Q)$. In particular, when two node variables are non-possibly equivalent, a deductive reasoning is carried out by applying equivalence and implication connectives between P and $wp(S, Q)$. We adopt a similar deduction for a node variable belonging to a concept complement and for a role complement. Using these deductions and the well-behaved wp properties, such as distributivity of conjunction and disjunction, we construct the formula P, which satisfies the implication $P \Rightarrow wp(S, Q)$ so that the triple $\{P\}S\{Q\}$ is always correct-by-construction.

V. AUTO-ACTIVE VERIFIER

Different techniques pertaining to formal program verification exist. An automatic verification requires no interaction with developers; a solver performs autonomously the formal verification of a program. However, it provides weak feedback in case of failure. Conversely, an interactive verification requires an expert to guide the proof assistant through its manipulations to perform the verification.

Since our purpose is to help a novice developer to achieve a correct transformation, we adopt an auto-active verification approach, which lies between automatic and interactive verification [12]. The auto-active approach expects a developer to annotate his code with specifications, then the verification will be done automatically. The process can be repeated in many iterations until the program is proved.

The auto-active approach has two main advantages: on the one hand it promotes an incremental development with rich feedback at each step, on the other hand it bridges the gap between a non-expert developer and a formal verification tool. This technique is used by AutoProof, part of EVE (Eiffel Verification Environment) to verify Eiffel programs [13], and by Dafny to verify functional correctness of Dafny programs [14].

In the next, an example of an incremental development for constructing a correct transformation program with the assistance of our auto-active verifier will be described.

A. *Incremental Developement*

Algorithm 1 shows an example of an incremental development using the auto-active approach to help developers construct specifications and refine the transformation triple. The idea is alternating the intervention of the static analyzer with the developer's in a progressive process until achieving finally a correct-by-construction triple.

Since the static analyzer calculates many conjunctions as preconditions with respect to a code and a postcondition as explained in Section IV, the number of extracted preconditions may be sizable. In this sense, using an interactive process to enable developers express more precisely their intention can help reduce the number of the extracted conjunctions given by the static analyzer.

```
Input: Code, Post
Output: Pre
repeat
    /* by Static Analyzer                          */
    E_Pre = extractedPre(Code, Post);
    /* by Developer                                */
    S_Pre = selectConjunctions(E_Pre);
    if isCorespondToIntention(S_Pre) then
        Pre = selectPrecondition(S_Pre);
    else
        refineCodeAndPost(Code, Post, S_Pre);
until isValid(Pre, Code, Post);
```

Algorithm 1: Example of an incremental development with Auto-active Verifier

First, the static analyzer suggests a precondition formula in the disjunctive normal form. Then the developer selects some of the suggested conjunctions that reflect his intention. If he is satisfied with the selection, the developer can take directly the chosen conjunctions as the final precondition and terminate the iteration. If not, the developer can refine his code or/and his postcondition to clarify his intention then starts a new iteration. In general, the developer can update his transformation code or refine his specification by injecting into them the facts of the chosen conjunctions. In this way, the transformation program is incrementally enhanced based on the developer's intention.

B. *Example*

In order to illustrate the application of our auto-active verifier in the proposed incremental process, we present in this section an example based on the development of an application simulating the activities of a hospital [22].

In this application, we consider $Patients$ and $Doctors$ in a hospital composed of several $Departments$. Each doctor works in (denoted by $worksIn$) a department and $treats$ patients. Each department is directed by (denoted by $directedBy$) a head who is one among its doctors. Each department is in charge of (denoted by $inChargeOf$) some diseases and registers the patients (denoted by $registers$) who $suffer$ one of those diseases. A hospitalized patient has a reference doctor (denoted by $refDr$) and is allocated (denoted by $allocated$) a $Room$.

Supposed that a graph is used to represent different concepts and individuals in the hospital. The following scenarios illustrate a development process of a novice developer to write a graph transformation for updating the hospital's status when patients arrive. We suppose that the developer does not write immediately a correct Hoare triple, but needs many iterations to refine his program with the static analyzer's help.

Rule *assignDoctor*

At the first step, the developer wants to write the rule $assignDoctor$ to assign a doctor dr to treating a patient p. He writes thus the rule's code to add the relations $treats$ and $refDr$ between the patient p and the doctor dr.

The chosen doctor dr has to work in the department dep where the patient p is registered. Moreover, a doctor who is head of a department can not treat more than 3 patients simultaneously. Considering these constraints, the developer writes the first version of his rule as shown in Figure 6.

```
rule assignDoctor {
  add(dr treats p);
  add(p refDr dr);
  post: p : Patient ∧ dr : Doctor
      ∧ dep : Department
      ∧ dep registers p ∧ dr worksIn dep
      ∧ head : Doctor ∧ dep directedBy head
      ∧ head : (≤ 3 treats Patient)
      ∧ dr treats p ∧ p : (≤ 1 refDr Doctor);
}
```

Figure 6. Rule $assignDoctor$ - First version

To complete the rule $assignDoctor$, the developer uses the static analyzer to extract a precondition. From the program in Figure 6, the static analyzer proposes the twelve following conjunctions as possible preconditions:

1) $p : Patient \land dr : Doctor \land dep : Department$
$\land\ dr\ worksIn\ dep \land dep\ registers\ p$
$\land\ head : Doctor \land dep\ directedBy\ head$
$\land\ head : (\leq 2\ treats\ Patient)$
$\land\ p : (\leq 0\ refDr\ Doctor)$

2) $p : Patient \land dr : Doctor \land dep : Department$
$\land\ dr\ worksIn\ dep \land dep\ registers\ p$
$\land\ head : Doctor \land dep\ directedBy\ head$
$\land\ head : (\leq 2\ treats\ Patient)$
$\land\ p\ !refDr\ dr$
$\land\ p : (\leq 0\ refDr\ Doctor)$

3) $p : Patient \land dr : Doctor \land dep : Department$
$\land\ dr\ worksIn\ dep \land dep\ registers\ p$
$\land\ head : Doctor \land dep\ directedBy\ head$
$\land\ head : (\leq 2\ treats\ Patient)$
$\land\ p\ refDr\ dr$
$\land\ p : (\leq 1\ refDr\ Doctor)$

4) $p : Patient \land dr : Doctor \land dep : Department$
$\land\ dr\ worksIn\ dep \land dep\ registers\ p$
$\land\ head : Doctor \land dep\ directedBy\ head$
$\land\ dr \neq head$
$\land\ head : (\leq 3\ treats\ Patient)$
$\land\ p : (\leq 0\ refDr\ Doctor)$

5) $p : Patient \land dr : Doctor \land dep : Department$
$\land\ dr\ worksIn\ dep \land dep\ registers\ p$
$\land\ head : Doctor \land dep\ directedBy\ head$
$\land\ dr\ treats\ p$
$\land\ head : (\leq 3\ treats\ Patient)$
$\land\ p : (\leq 0\ refDr\ Doctor)$

6) $p : Patient \land dr : Doctor \land dep : Department$
$\land\ dr\ worksIn\ dep \land dep\ registers\ p$
$\land\ head : Doctor \land dep\ directedBy\ head$
$\land\ dr\ !treats\ p$
$\land\ dr = head$
$\land\ head : (\leq 2\ treats\ Patient)$
$\land\ p : (\leq 0\ refDr\ Doctor)$

7) $p : Patient \land dr : Doctor \land dep : Department$
$\land\ dr\ worksIn\ dep \land dep\ registers\ p$
$\land\ head : Doctor \land dep\ directedBy\ head$
$\land\ dr\ treats\ p$
$\land\ p\ !refDr\ dr$
$\land\ head : (\leq 3\ treats\ Patient)$
$\land\ p : (\leq 0\ refDr\ Doctor)$

8) $p : Patient \land dr : Doctor \land dep : Department$
$\land\ dr\ worksIn\ dep \land dep\ registers\ p$
$\land\ head : Doctor \land dep\ directedBy\ head$
$\land\ dr \neq head$
$\land\ p\ !refDr\ dr$
$\land\ head : (\leq 3\ treats\ Patient)$
$\land\ p : (\leq 0\ refDr\ Doctor)$

9) $p : Patient \land dr : Doctor \land dep : Department$
$\land\ dr\ worksIn\ dep \land dep\ registers\ p$
$\land\ head : Doctor \land dep\ directedBy\ head$
$\land\ dr \neq head$
$\land\ p\ refDr\ dr$
$\land\ head : (\leq 3\ treats\ Patient)$
$\land\ p : (\leq 1\ refDr\ Doctor)$

10) $p : Patient \land dr : Doctor \land dep : Department$
$\land\ dr\ worksIn\ dep \land dep\ registers\ p$
$\land\ head : Doctor \land dep\ directedBy\ head$
$\land\ dr\ treats\ p$
$\land\ p\ refDr\ dr$
$\land\ head : (\leq 3\ treats\ Patient)$
$\land\ p : (\leq 1\ refDr\ Doctor)$

11) $p : Patient \land dr : Doctor \land dep : Department$
$\land\ dr\ worksIn\ dep \land dep\ registers\ p$
$\land\ head : Doctor \land dep\ directedBy\ head$
$\land\ dr\ !treats\ p \land p\ refDr\ dr$
$\land\ dr = head$
$\land\ head : (\leq 2\ treats\ Patient)$
$\land\ p : (\leq 1\ refDr\ Doctor)$

12) $p : Patient \wedge dr : Doctor \wedge dep : Department$
$\wedge\ dr\ worksIn\ dep \wedge dep\ registers\ p$
$\wedge\ head : Doctor \wedge dep\ directedBy\ head$
$\wedge\ dr\ !treats\ p$
$\wedge\ p\ !refDr\ dr$
$\wedge\ dr\ =\ head$
$\wedge\ head : (\leq 2\ treats\ Patient)$
$\wedge\ p : (\leq 0\ refDr\ Doctor)$

Among these conjunctions, eight formulae contain the fact $p : (\leq 0\ refDr\ Doctor)$ stating that the examined patient p does not have yet a reference doctor. This fact reflects the developer's precondition regarding a patient, thus first he selects the formulae 1, 2, 4, 5, 6, 7, 8, 12 as possible preconditions.

Now, to treat a patient p, the developer wants to choose a doctor who is not in charge of the department. To clarify his intention and help the static analyzer reduce the number of extracted conjunctions, the developer injects a $select$ statement into his code in order to choose the instance dr that is distinct from the instance $head$. Figure 7 shows the modified body of the rule $assignDoctor$.

```
select dr with
        dr : Doctor ∧ dr worksIn dep ∧ dr ≠ head;
add(dr treats p);
add(p refDr dr);
```

Figure 7. Refined code of the rule $assignDoctor$

Using again the static analyzer with this new code, only one precondition is extracted. It is taken directly by the developer as the precondition of his rule. The final complete rule $assignDoctor$ is shown in Figure 8.

```
rule assignDoctor {
  pre: p : Patient ∧ dep : Department
       ∧ dep registers p ∧ head : Doctor
       ∧ dep directedBy head
       ∧ head : (≤ 3 treats Patient)
       ∧ p : (≤ 0 refDr Doctor);

  select dr with
      dr : Doctor ∧ dr worksIn dep ∧ dr ≠ head;
  add(dr treats p);
  add(p refDr dr);

  post: p : Patient ∧ dr : Doctor
       ∧ dep : Department
       ∧ dep registers p ∧ dr worksIn dep
       ∧ head : Doctor ∧ dep directedBy head
       ∧ head : (≤ 3 treats Patient)
       ∧ dr treats p ∧ p : (≤ 1 refDr Doctor);
}
```

Figure 8. Rule $assignDoctor$ - Final version

Rule *registerPatient*

The rule $assignDoctor$ assumes that the examined patient is already registered in a department. This means that before calling $assignDoctor$ another rule is needed to register the patient. For this purpose, the developer now writes the rule $registerPatient$ to receive and place a patient in an appropriate department according to his illness.

Because $registerPatient$ precedes $assignDoctor$, the postcondition of the rule $registerPatient$ must include the facts regarding $Patient$ and $Department$ concepts in the precondition of the rule $assignDoctor$, i.e., the facts $p : Patient$, $dep : Department$ and $dep\ registers\ p$.

To make sure that a patient is registered in a department that is in charge of his disease, the developer adds to the $registerPatient$'s postcondition the following two facts: $p\ suffers\ disease$ and $dep\ inChargeOf\ disease$.

Since the person p was not considered as a patient before being registered, in the rule's code, the developer writes the statements to declare p belongs to the concept $Patient$ and to create a relationship between the department dep and the patient p. Figure 9 shows the first version of the rule $registerPatient$.

```
rule registerPatient {
  add(p : Patient);
  add(dep registers p);

  post: p : Patient ∧ dep : Department
       ∧ dep registers p
       ∧ p suffers disease
       ∧ dep inChargeOf disease;
}
```

Figure 9. Rule $registerPatient$ - First version

Taking as input the program in Figure 9, the static analyzer extracts the following precondition:

$$dep : Department \wedge dep\ inChargeOf\ disease \wedge p\ suffers\ disease$$

The developer takes the extracted precondition and add to it the fact $p : Person \cap !Patient$ specifying that before applying the rule p is of concept $Person$ and not of type $Patient$. Figure 10 shows the final version of the rule $registerPatient$.

```
rule registerPatient {
  pre: dep : Department
       ∧ dep inChargeOf disease
       ∧ p : Person ∩ !Patient ∧ p suffers disease;

  add(p : Patient);
  add(dep registers p);

  post: p : Patient ∧ dep : Department
       ∧ dep inChargeOf disease
       ∧ p suffers disease ∧ dep registers p;
}
```

Figure 10. Rule $registerPatient$ - Final version

With the help of the static analyzer, the developer can develop two correct-by-construction rules as we've just presented. Now, he can write the main program of his application as followed to manage all people who have arrived at the hospital as shown in 11. In this code, the operation ! allows applying each rule iteratively as long as possible.

```
main {
    receivePatient!;
    assignDoctor!;
}
```

Figure 11. Program *ManagePatients*

VI. Integrated Development Environment

Aiming at integrating various tools to assist in developing and reasoning about graph transformations, the static analyzer is part of an experimental environment that provides the assistance in coding, executing and verifying transformations written in Small-t$\mathcal{ALC}$ [11].

Figure 12 shows the big picture of our framework and its components. Each component provides a specific support for Small-t$\mathcal{ALC}$ programs: the compiler translates a Small-t$\mathcal{ALC}$ program to an executable code; the dynamic analyzer examines the behavior of a running program; the static analyzer helps achieve correct transformations and the prover verifies the correctness of programs. The development of each component is based on an implementation of Small-t$\mathcal{ALC}$'s semantics in an appropriate foundation.

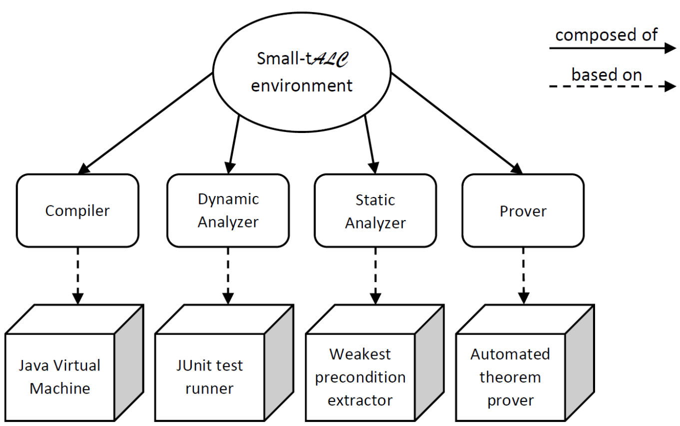


Figure 12. Overview of the Small-t$\mathcal{ALC}$ environment

The compiler, developed using the compiler generator Coco/R, produce the byte code of a Small-t$\mathcal{ALC}$ program for the Java Virtual Machine. This executable code transforms a source graph into a target graph.

The dynamic analyzer can be used to find inconsistencies between a transformation code and its specifications by executing the transformation generated by the compiler then applying automated tests on the target graph. The test cases are generated from the postcondition using our Small-t$\mathcal{ALC}$ testing library, which is based on JUnit assertions. The input graph can be generated automatically from the precondition or can be given by the user.

The prover is a formal verification tool that verifies a transformation program with respect to its pre- and postconditions by translating it into Isabelle/HOL logic and generating verification conditions. These verification conditions are passed to an automated theorem prover, which can then formally prove the correctness of the code. In case of failure, the prover displays a counter-example, which is a model of the precondition that does not satisfy the postcondition when applying the transformation.

In the following we illustrate the use of different tools in our integrated development environment via a scenario of writing a third rule in the application to manage a hospital.

Rule *allocateRoom*

Suppose that the developer now wants to write a rule to allocate a hospital's room r ($r : Room$) to a patient p ($p : Patient$) who is already assigned to a doctor dr ($dr\ treats\ p$) but is not allocated a room yet ($p : (\leq 0\ allocated\ Room)$).

The allocated room must be available ($r : AvailableRoom$) and in the department dep where p is registered ($p\ allocated\ r \wedge r\ isIn\ dep \wedge dep\ registers\ p$). The developer then writes the first version of the rule *allocateRoom* as shown in Figure 13.

```
rule allocateRoom {
  pre: p : Patient ∧ r : Room ∩ AvailableRoom
       ∧ r isIn dep ∧ dep registers p
       ∧ dr treats p
       ∧ p : (≤ 0 allocated Room);

  add(p allocated r);
  delete(r : AvailableRoom);

  post: p : Patient ∧ r : Room ∩ !AvailableRoom
       ∧ p allocated r
       ∧ p : (≤ 0 allocated Room);
}
```

Figure 13. Rule *allocateRoom* - First version

When submitting the program in Figure 13 to the prover, the proof fails and a counter-example is given as shown in Figure 14.

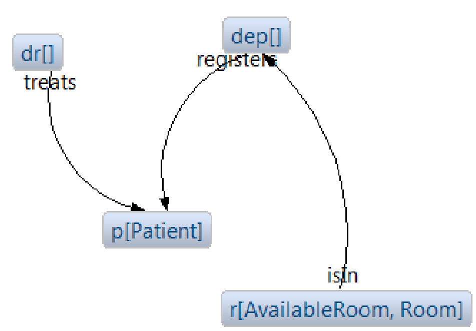


Figure 14. A counter-example of the rule *allocateRoom*

Having difficulties to locate the error from the counter-example given by the prover, the developer uses the dynamic analyzer to examine his program by running tests. From the postcondition, the dynamic analyzer generates the following test cases:

1) $assertExistNode(graph, p, AtomicConcept(Patient));$

2) $assertExistNode(graph, r, AtomicConcept(Room));$

3) $assertNotExistNode(graph, r, AtomicConcept(AvailableRoom));$

4) $assertExistEdge(graph, p, allocated, r);$

5) $assertAtMostNumberEdges(graph, p, allocated, AtomicConcept(Room), 0);$

When applying the rule $allocatedRoom$ on a generated graph source from the precondition then executing the tests on the target graph , the first four test cases succeed but the fifth test fails. The failure of this $assertAtMostNumberEdges$ test means that the target graph does not respect the condition "there is at most 0 relation $allocated$ between p and the individuals of the type $Room$". The error is therefore from the fact $p : (\leq 0\ allocated\ Room)$ in the given postcondition.

Thanks to this diagnosis, the developer notices that it is necessary to increase the number of restrictions in the erroneous fact as follows : $p : (\leq 1\ allocated\ Room)$. Now he resubmits the modified program to the prover, which indicates that the triplet is correct.

Our Small-t$\mathcal{ALC}$ environment provides different levels of assistance in writing both rule's statements and their specifications. We choose a testing framework as infrastructure of the dynamic analyzer for providing immediate feedback and detailed diagnostics to help correct rule code with respect to given specifications. On the other hand, the static analyzer helps in an auto-active approach construct a correct-by-construction transformation given a postcondition and partially a code. Consequently, both produce a valid Hoare triple of a rule to be eventually proved formally by the prover.

VII. Related work

Most of the logic-based approaches for graph transformations focus on the verification question. Thus, they attempt to encode graph conditions in an appropriate logic that is both expressive and decidable. The work in [23] expresses invariants in Computation Tree Logic (CTL). Becker et al. [24] encoded graph patterns as first-order predicates and created symbolic representations for possible violations of the rule's properties. Inaba et al. [25] verified graph transformations against the graph structural constraints in Monadic Second-Order logic (MSO). Calvanese et al. [26] studied the static verification for evolving graph databases where the integrity constraints are expressed in DL $\mathcal{ALCHOIQ}$. Brenas et al. [27] provided a decidable logic based on the DL $\mathcal{SROIQ}$ for reasoning on program processing structures defined as graphs. Like us, Selim et al. [28] proposed a direct verification framework for their transformation language DSLTrans so that no intermediate representation for a specific proving framework is required. They used symbolic execution to build a finite set of path conditions representing all transformation executions through a formal abstraction relation and thus allow formal properties to be exhaustively proved. Their property language based on graph patterns and propositional logic proposes a limited expressiveness and the property-proving algorithm was presented as a proof-of-concept.

The works in [24] and [29] share with ours some ideas with respect to the assistance in producing a Hoare triple. Becker et al. [24] proposed an iterative development of consistency-preserving refactorings, which are specified in a rule-based manner and rely on a graph-transformation formalization. Given a modeling language with well-formedness constraints and a refactoring specification, Becker et al. [24] use an invariant checker to detect and report constraint violations via counter-examples and lets developers modify their refactoring iteratively. Similarly to us, Clariso et al. [29] used backward reasoning to automatically synthesize application conditions for model transformation rules. Application conditions are derived from the OCL expression representing the rules's postconditions and the atomic rewriting actions performed by the rule. However, OCL expressions are not really suitable for exploring the graph properties of the underlying model structures. It is thus rather cumbersome when used for verifying complex model transformations. To obtain a higher abstraction and benefit the formal verification of graph rewriting systems, recently, there are some graph-based approaches proposing the translations of a set of OCL expressions to graph patterns [30] or nested graph constraints [31].

VIII. Conclusion and future work

The distinctive feature of Small-t$\mathcal{ALC}$ is that it uses the same logic $\mathcal{ALCQ}$ to represent graphs, to code a transformation and to reason about graph transformations in a Hoare style. In order to assist users in developing correct transformations, we propose a fine analysis of the weakest precondition to take into account special situations of a program on the basis of an alias calculus. Our auto-active approach allows developers to select a precondition to annotate their code according to their intention.

It would be interesting in our framework to automatically infer and test invariant candidates for loop constructs gathered from their corresponding postcondition as proposed in [32]. This attempt is based on the fact that a Small-t$\mathcal{ALC}$ loop often iterates on individuals selected from a logic formula in order to achieve the same property for the transformed elements.

As a complement to a Hoare triple verification, we are working on effects of rules execution in terms of DL reasoning services at the specification rule level. A Small-t$\mathcal{ALC}$ rule execution updates a knowledge base founded upon a finite set of $\mathcal{ALCQ}$ concept inclusions (TBox) and a finite set of $\mathcal{ALCQ}$ concept and role assertions (ABox). This leads to a reasoning problem about a knowledge base consistency embodied by a graph in Small-t$\mathcal{ALC}$ [33].

Acknowledgment

Part of this research has been supported by the *Climt* (Categorical and Logical Methods in Model Transformation) project (ANR-11-BS02-016). We are grateful for Martin Strecker for the interesting discussions.

References

[1] A. Makhlouf, C. Percebois, and H. N. Tran, "A Precondition Calculus for Correct-by-Construction Graph Transformations," in International Conference on Software Engineering Advances (ICSEA), Athens (Greece), 08/10/2017-12/10/2017. http://www.iaria.org/: IARIA, 2017, pp. 172–177.

[2] A. Habel and K.-H. Pennemann, "Correctness of high-level transformation systems relative to nested conditions," Mathematical. Structures in Comp. Sci., vol. 19, no. 2, Apr. 2009, pp. 245–296.

[3] A. Rensink, "Representing first-order logic using graphs," in Graph Transformations: Second International Conference ICGT,. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 319–335.

[4] B. Courcelle, "Handbook of theoretical computer science (vol. b)." Cambridge, MA, USA: MIT Press, 1990, ch. Graph Rewriting: An Algebraic and Logic Approach, pp. 193–242.

[5] K.-H. Pennemann, "Resolution-like theorem proving for high-level conditions," in Graph Transformations: 4th International Conference, ICGT. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 289–304.

[6] F. Orejas, H. Ehrig, and U. Prange, "A logic of graph constraints," in Fundamental Approaches to Software Engineering: 11th International Conference, FASE. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 179–198.

[7] L. Lambers and F. Orejas, "Tableau-based reasoning for graph properties," in Graph Transformation: 7th International Conference, ICGT. Cham: Springer International Publishing, 2014, pp. 17–32.

[8] A. Habel and D. Plump, "M,N-adhesive transformation systems," in Proceedings of the 6th International Conference on Graph Transformations, ser. ICGT'12. Berlin, Heidelberg: Springer-Verlag, 2012, pp. 218–233.

[9] D. Plump, "The graph programming language gp," in Algebraic Informatics: Third International Conference, CAI. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 99–122.

[10] M. Strecker, "Modeling and verifying graph transformations in proof assistants," Electron. Notes Theor. Comput. Sci., vol. 203, no. 1, Mar. 2008, pp. 135–148.

[11] N. Baklanova, J. H. Brenas, R. Echahed, A. Makhlouf, C. Percebois, M. Strecker, and H. N. Tran, "Coding, executing and verifying graph transformations with small-t$\mathcal{ALCQ}$e," in 7th Int. Workshop on Graph Computation Models (GCM), 2016, URL: http://gcm2016.inf.uni-due.de/ [accessed: 2018-05-06].

[12] K. R. M. Leino and N. Polikarpova, "Verified calculations," in Verified Software: Theories, Tools, Experiments: 5th International Conference, VSTTE. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014, pp. 170–190.

[13] C. A. Furia, M. Nordio, N. Polikarpova, and J. Tschannen, "Autoproof: auto-active functional verification of object-oriented programs," International Journal on Software Tools for Technology Transfer, 2016, pp. 1–20.

[14] K. R. M. Leino, "Dafny: An automatic program verifier for functional correctness," in Proceedings of the 16th International Conference on Logic for Programming, Artificial Intelligence, and Reasoning, ser. LPAR'10. Berlin, Heidelberg: Springer-Verlag, 2010, pp. 348–370.

[15] A. Makhlouf, H. N. Tran, C. Percebois, and M. Strecker, "Combining dynamic and static analysis to help develop correct graph transformations," in Tests and Proofs: 10th International Conference, TAP. Switzerland: Springer International Publishing, 2016, pp. 183–190.

[16] M. Nagl, "Set theoretic approaches to graph grammars," in Proceedings of the 3rd International Workshop on Graph-Grammars and Their Application to Computer Science. London, UK, UK: Springer-Verlag, 1987, pp. 41–54.

[17] M. Krötzsch, F. Simancik, and I. Horrocks, "A description logic primer," arXiv preprint arXiv:1201.4089, 2012, URL: http://arxiv.org/abs/1201.4089 [accessed: 2018-05-06].

[18] M. Schmidt-Schauß and G. Smolka, "Attributive concept descriptions with complements," Artif. Intell., vol. 48, no. 1, Feb. 1991, pp. 1–26.

[19] E. W. Dijkstra and C. S. Scholten, Predicate Calculus and Program Semantics. New York, NY, USA: Springer-Verlag New York, Inc., 1990.

[20] J. H. Brenas, R. Echahed, and M. Strecker, "On the closure of description logics under substitutions," in Proceedings of the 29th International Workshop on Description Logics, Cape Town, South Africa, April 22-25, 2016., M. Lenzerini and R. Peñaloza, Eds., 2016, URL: http://ceur-ws.org/Vol-1577/paper_47.pdf [accessed: 2018-05-06].

[21] B. Meyer, "Steps towards a theory and calculus of aliasing," Int. J. Software and Informatics, vol. 5, no. 1-2, 2011, pp. 77–115.

[22] J. H. Brenas, R. Echahed, and M. Strecker, "Ensuring correctness of model transformations while remaining decidable," in Theoretical Aspects of Computing – ICTAC 2016, A. Sampaio and F. Wang, Eds. Cham: Springer International Publishing, 2016, pp. 315–332.

[23] Z. Langari and R. Trefler, "Application of graph transformation in verification of dynamic systems," in Integrated Formal Methods: 7th International Conference, IFM. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 261–276.

[24] B. Becker, L. Lambers, J. Dyck, S. Birth, and H. Giese, "Iterative development of consistency-preserving rule-based refactorings," in Theory and Practice of Model Transformations: 4th International Conference, ICMT. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 123–137.

[25] K. Inaba, S. Hidaka, Z. Hu, H. Kato, and K. Nakano, "Graph-transformation verification using monadic second-order logic," in Proceeding of the 13th International ACM SIGPLAN Symposium on Symposium on Principles and Practice of Declarative Programming. ACM Press, Jul. 2011.

[26] D. Calvanese, M. Ortiz, and M. Simkus, "Evolving graph databases under description logic constraints," in Proc. of the 26th Int. Workshop on Description Logics (DL 2013), 2013.

[27] J. H. Brenas, R. Echahed, and M. Strecker, "A hoare-like calculus using the SROIQ σ logic on transformations of graphs," in Theoretical Computer Science - 8th IFIP TC 1/WG 2.2 International Conference, TCS, 2014, pp. 164–178.

[28] G. M. Selim, L. Lúcio, J. R. Cordy, J. Dingel, and B. J. Oakes, "Specification and verification of graph-based model transformation properties," in International Conference on Graph Transformation. Springer, 2014, pp. 113–129.

[29] R. Clarisó, J. Cabot, E. Guerra, and J. de Lara, "Backwards reasoning for model transformations," J. Syst. Softw., vol. 116, no. C, Jun. 2016, pp. 113–132.

[30] G. Bergmann, "Translating ocl to graph patterns," in Model-Driven Engineering Languages and Systems: 17th International Conference, MODELS. Cham: Springer International Publishing, 2014, pp. 670–686.

[31] H. Radke, T. Arendt, J. S. Becker, A. Habel, and G. Taentzer, "Translating essential ocl invariants to nested graph constraints focusing on set operations," in Graph Transformation: 8th International Conference, ICGT. Cham: Springer International Publishing, 2015, pp. 155–170.

[32] J. Zhai, H. Wang, and J. Zhao, "Post-condition-directed invariant inference for loops over data structures," in Proceedings of the 2014 IEEE Eighth International Conference on Software Security and Reliability-Companion, ser. SERE-C '14. Washington, DC, USA: IEEE Computer Society, 2014, pp. 204–212.

[33] U. Sattler, "Reasoning in description logics: Basics, extensions, and relatives," in Reasoning Web: Third International Summer School. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 154–182.

Scavenging Run-time Resources to Boost Utilization in Component-based Embedded Systems with GPUs

Gabriel Campeanu, Saad Mubeen
Mälardalen Real-Time Research Center (MRTC),
Mälardalen University, Västerås, Sweden
Email: {gabriel.campeanu, saad.mubeen}@mdh.se

Abstract—**Many modern embedded systems with GPUs are required to process huge amount of data that is sensed from their environment. However, due to some inherent properties of these systems such as limited energy, computation and storage resources, it is important that the resources should be used in an efficient way. For example, camera sensors of a robot may provide low-resolution frames for positioning itself in an open environment and high-resolution frames to analyze detected objects. In this paper, we introduce a method that, when possible, scavenges the unused resources (i.e., memory and number of GPU computation threads) from the critical functionality and distributes them to the non-critical functionality. As a result, the overall system performance is improved without compromising the critical functionality. The method uses a monitoring solution that checks the utilization of the system resources and triggers their distribution to the non-critical functionality whenever possible. As a proof of concept, we realize the proposed method in a state-of-the-practice component model for embedded systems. As an evaluation, we use an underwater robot case study to evaluate the feasibility of the proposed solution.**

Keywords–GPU; embedded system; component-based software development; CBD; model-based development; MBD; resource utilization; monitor.

I. INTRODUCTION

This paper substantially extends the authors' previous work [1]. Embedded systems are found in almost all contemporary electronic products. Their applications range from simple and small-sized products, e.g. a wireless controlled toy car or an airplane to very complex and large-sized systems such as premium cars and airplanes. Many modern embedded systems are developed to process huge amount of data that is originated from the interaction with their environments. For example, the Google autonomous car processes around 750 MB of data per second [2]. Massive computing power and parallel execution of software is required to process such a large amount of data. The traditional embedded systems are unable to handle these data-intensive applications, mainly due to reduced computational power and support for parallel execution of software.

Graphics Processing Units (GPUs) offer a promising solution to deal with this challenge. A GPU supports a parallel execution model, which allows multiple data to be processed in parallel. However, a GPU cannot be used in isolation, i.e., the GPU needs a Central Processing Unit (CPU) to activate the activities (threads) that are intended to run in parallel on it. Thanks to the recent advances in the semiconductor and chip industry, there are several vendors that manufacture heterogeneous computing platforms which contain a combination of CPUs, GPUs and other computation resources on the same board. For example, vendors such as NVIDIA, AMD and Samsung provide their own embedded heterogeneous solutions on the same board such as NVIDIA Jetson TK1 [3], AMD R-464L [4] and Samsung Exynos 8 [5] respectively.

The amount of data captured by an embedded system from its environment can significantly impact the management of its resources (e.g., memory and computation power), which in turn can impact its performance. One way to optimize the resource usage is to collect variable stream-size of data from the sensors depending upon different situations. For example, the ProcImage500-Eagle camera sensor [6] can be configured to capture low- or high-resolution frames depending upon the environment. For example, a robot fitted with such a camera may use low-resolution data frames to examine its position in an open environment. On the other hand, the robot may use high-resolution frames to inspect the target objects in a detailed manner. The high-resolution frames require larger memory footprints and more computation power (and energy) to be processed by the GPU. Whereas, the low-resolution frames are delivered with faster frame rate, occupy less memory and require lower computation power for GPU processing.

The system resources in many embedded systems are shared between non-critical and critical functions. The non-critical functions are not constrained by any resource requirements. Hence, these functions are expected to provide the best-effort service. Whereas, the critical functions are constrained by resource requirements, often stringent, that must be met during the execution of the system. Hence, it is ensured that all the system resources that are required by the critical functions are always available to them. For example, a vision system of a robot represents a critical function. This system is designed in such a way that it is always guaranteed enough resources to process the high-resolution frames. Even when the cameras provide low-resolution frames, the system still occupies the same amount of resources as if it were processing the high-resolution frames. As a consequence, the system resources are wasted when they are not needed by the critical function. We argue that the non-critical functions can benefit from the resources of the critical functions during the intervals when they are not used. For example, when the robot utilizes low-resolution frames, a non-critical function such as a logger system can benefit from the extra memory which is not being used by the vision system.

This paper provides a method to automatically compute the unused resources in the critical part of the system. The method then distributes the computed resources to the non-critical parts of the system. The proposed method is based on a run-time monitoring engine that monitors the critical part of the system to detect any changes in its resource requirements. If the engine detects a reduction in the resource requirements of the critical system, it triggers the proposed method to calculate the unused memory based on the actual resource usage by the critical part of the system. The information regarding the amount of available memory is provided to non-critical part of the system, which can benefit from the available extra resource.

The initial conference paper [1] presented a method to improve the resource utilization in embedded systems based on the system memory as the only run-time resource. The submitted journal paper extends our method by considering other run-time resources such as the GPU computation threads together with the system memory to further boost the resource utilization of embedded systems with GPUs.

The rest of the paper is organized as follows. Section II describes the background context. Section III formulates the problem and describes it with the help of a case study. The overview of our solution is described in Section IV-A and its realization is presented in Section IV-B. Section IV-C discusses the implementation of the solution. The evaluation of our method applied to the case study is discussed in Section V. Finally, Section VI introduces the existing work related to our approach, while Section VII concludes the paper.

II. Background

The section provides background information on GPUs and some software development strategies for embedded systems.

A. GPUs

GPUs were developed in 90s and were employed only in graphic-based applications. With the passage of time GPUs became more popular due to increase in their computation power and ease of use. As a result, GPUs have been utilized in different types of applications, even becoming the general-purpose processing units referred to as GPGPUs [7]. For example, cryptography applications [8] and Monte Carlo simulations [9] use GPU-based solutions. Equipped with a parallel architecture, the GPU may employ thousands of computation threads at a time through its multiple cores. Compared to the traditional CPU, the GPU delivers an improved performance with respect to processing multiple data in parallel. For example, simulation of bio-molecular systems have achieved 20 times speed-up on GPU [10].

One of the GPU characteristics is that it cannot function without the help of a CPU. The CPU is considered as the brain of the system that triggers all the activities related to GPU, such as the execution of functionality onto GPU. The GPU often has its own private memory system, which requires data to be copied from one memory system to the other in order to be accessed by the corresponding processing unit. Note that even if the CPU and GPU are integrated on the same physical chip, the chip's memory is divided in two distinct memory spaces, one for each processing unit. Figure 1 presents an example of an embedded platform that has the CPU and GPU integrated on the same chip. While the CPU is equipped with two cores (i.e., two execution threads), the GPU has several cores characterized by hundred of execution threads. The physical memory is divided in two sections, each processing unit accessing its private memory section. Due to the reduced physical size, lower energy usage and costs, this embedded solution is one of the commonly used solution in the industry. These reasons motivate us to consider the platform that has distinct memory spaces for CPU and GPU.

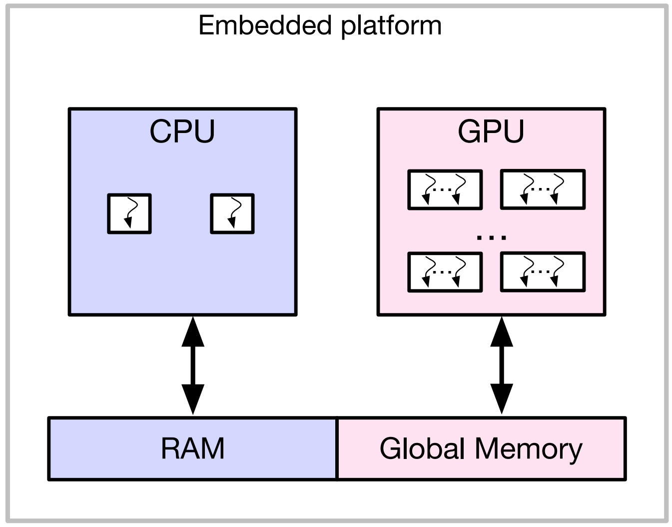


Figure 1. A CPU-GPU embedded platform.

B. Model- and Component-based Software Development

The software complexity in embedded systems has significantly increased in various domains, e.g., the automotive and robotic domains. The software development techniques that are based on the principles of model-based engineering (MBE) and component-based software engineering (CBSE) have proven efficient in dealing with the software complexity [11], [12]. CBSE is also successfully employed to develop robotic applications [13]. Using these paradigms, models are used throughout the development process. These software models allow the development of applications by connecting software units, called *software components*. CBSE promotes the (re-)use of the same component in different contexts. One benefit of adopting CBSE is the development efficiency that is achieved through reusable software components. A key concept of CBD is the *encapsulation*, where all the information of a component is encapsulated inside, hidden from anything outside. A way to access the encapsulated information is through *interfaces*. In this work, we focus on *port-based interfaces*, where the ports are access points of software components. MBE and CBSE have been successfully adopted by the industry through various component models.

When a software component is developed, the specifications of a component model are followed. For example, Component Object Model (COM) specifies that all of the COM components should be constructed with an *IUnknown* interface [14]. The component model also describes the way its components interact and how they are combined in systems. There exists many component models, some designed for particular domains (e.g., automotive) and other built on specific technological platforms (e.g., Enterprise Java Beans [15]). Note that component models follow various interaction styles that are suitable for different types of applications [16]. We mention the *request-response* and *sender-receiver* interaction styles that are utilized in AUTOSAR [17] component model when developing automotive applications. Another style utilized by e.g., Rubus Component Model (RCM) [18] and IEC 61131 [19] component models, is the *pipe-and-filter* interaction style. This particular style is characteristic to streaming-of-event type of applications and allows an easy mapping between the flow of system actions and control specifications, characteristic to real-time and safety-critical applications.

The existing component models that can be used to build stream-of-event applications, e.g., RCM, AUTOSAR, IEC 61131 and ProCom[20], face the challenge of dealing with (streaming) data that can change its memory footprint on-the-fly. For example, RCM defines that its components use the same fixed memory footprint throughout the execution of the application. In order to ensure the required resources to the critical functionality, resources are assigned to each RCM software component, with respect to its worst-case resource demand for the entire system execution. Therefore, RCM and similar component models (discussed above) do not support any mechanism to release the resources when they are not required (by the critical part of the system).

In this paper we consider RCM as a proof of concept for the implementation of our solution. A Rubus software component, also known as the *software circuit*, is the lowest-level hierarchical element. It is characterized by input ports and output ports. A feature of RCM is that there is a clear separation between the control flow and data flow. Therefore, a software circuit has two types of ports, i.e., data and trigger ports. Figure 2 presents three connected Rubus components, each equipped with one single (input and output) trigger port, and one or several (input and output) data ports. The software circuit uses the read-execute-write execution semantics. Initially in an idle state, a component is activated through its input trigger port. It starts by reading the data from its input data ports, then it executes its functionality, followed by writing the results to the output data ports. Finally, it transfers the control through its output trigger port and re-enters the idle state.

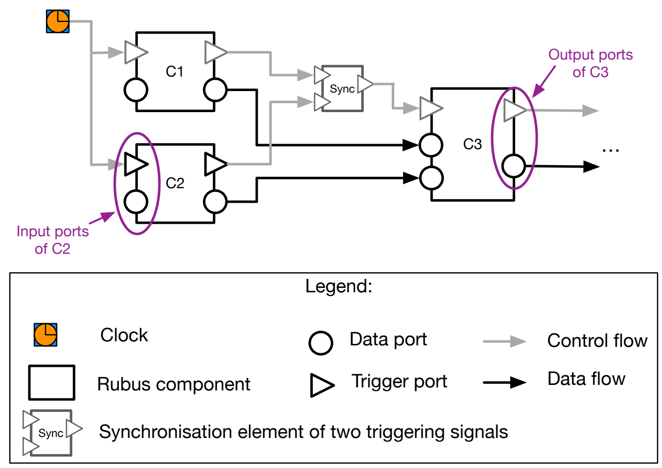


Figure 2. Three connected Rubus software components.

III. Problem Description

The run-time resources and energy usage in an embedded system can be reduced by decreasing the data produced by sensors with respect to the changing conditions in its environment. For example, a robot may require low-resolution frames to process open-space environments but may utilize high-resolution frames when analyzing close ups of detected objects. Therefore, the robot cameras may be set to provide, on-the-fly, frames with different resolutions based on, e.g., distance to the tracked objects. However, due to the rules set by the existing component models for the construction of software components, the size of a component's input data is fixed during the execution of the system. One way to ensure the guaranteed execution of the system is to allocate the system resources to software components, at the design time, to deal with the maximum footprint of data produced by sensors. For example, if a camera produces frames with 1280 x 1024 pixels, the software components that process the camera feedback utilize memory corresponding to the camera's frames. Even when the camera produces lower quality frames (e.g., 640 x 480 pixels) with a lower memory footprint, the software components are set to utilize the memory footprint characteristic to 1280 x 1024 pixel frames, resulting in underutilization of the system memory.

We use a case study as a running example to discuss the problem in detail. The case study is centered around an underwater robot that autonomously navigates under water, fulfilling various missions (e.g., tracking red buoys) [21]. The robot contains a CPU-GPU embedded board that is connected to various sensors (e.g., cameras) and actuators (e.g., thrusters). Sensors provide a continuous flow of environment data that is processed by the GPU on-the-fly. A simplified component-based software architecture of the robot's vision system is depicted in Figure 3. The software architecture, realized using RCM, contains nine software components. The *Camera1* and *Camera2* software components are connected to the physical sensors and convert the received data into readable frames. The

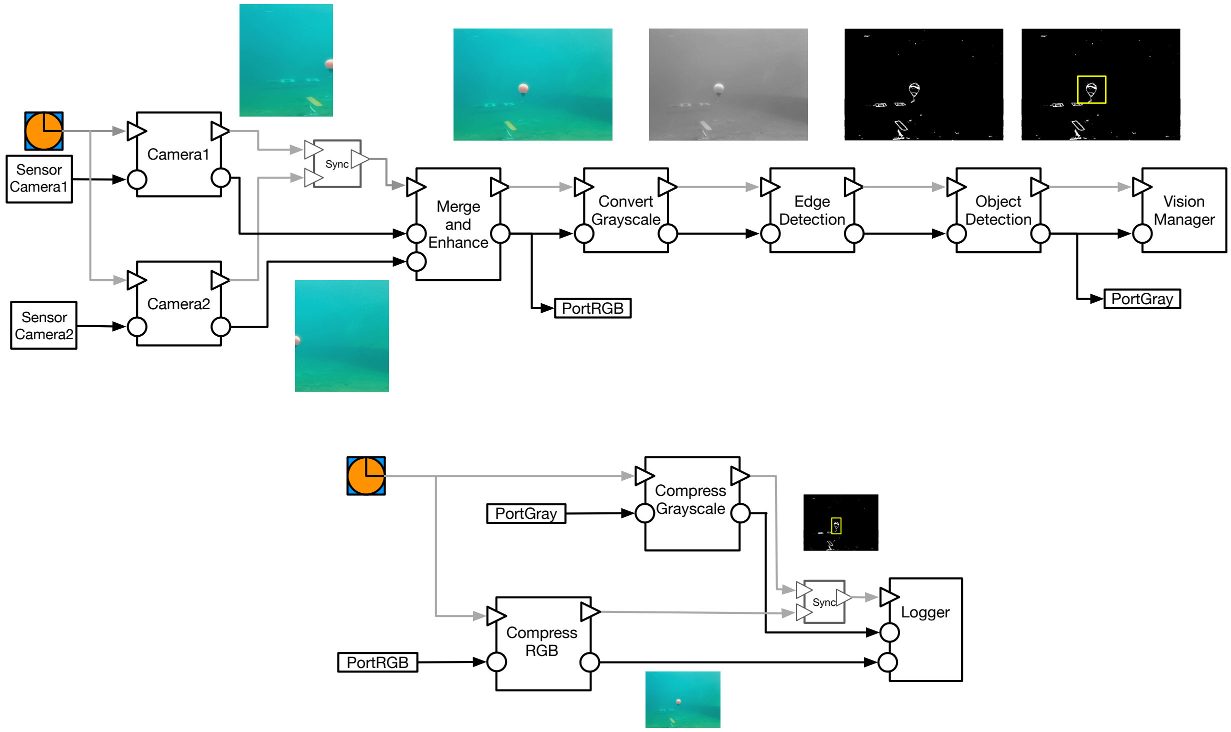


Figure 3. Component-based Rubus vision system of the underwater robot.

MergeAndEnhance software component reduces the noise and merges the two frames using the GPU. The resulted frame is converted into a gray-scale frame by *ConvertGrayscale* software component (on the GPU), which is forwarded to *EdgeDetection* software component that produces a black-and-white frame with detected edges. The *ObjectDetection* software component identifies the target object from the received frame and forwards the result to the system manager that takes appropriate actions, such as grabbing the detected objects. Due to the specifications of the functionalities (i.e., processing image), *MergeAndEnhance*, *ConvertGrayscale*, *EdgeDetection* and *ObjectDetection* components use the GPU to execute their behavior.

When the robot navigates underwater, the cameras are set to produce 640 x 480 pixel frames to track points for positioning itself. Due to the particularities of the water, sometime being muddy or the underwater vision being influenced by the weather conditions (e.g., cloudy, sunny), there is no need for high-resolution frames as the visibility is reduced. Figure 3 presents 640 x 480 pixel frames that contain several objects. While one of the missions is to track and touch buoys, the robot navigates to the detected objects. When the robot is close (e.g., one meter away) to the detected object, it requires high-resolution frames to observe and refine the details needed for the distinction between similar type of objects. In this case, cameras produce 1280 x 960 pixel frames.

Following the specifications of RCM, each software component is equipped with a constructor and a destructor. The constructor is executed once before the system run. Whereas, the destructor is executed when the system is properly switched off or reset. The constructor has the role to allocate resources needed by the software component, such as memory required by the internal behavior and output data ports. As it is executed only once, the constructor allocates a fixed memory size for the duration of entire execution life of the software component. For the presented vision system, the constructor of each software component reserves memory to handle e.g., input data of maximum size. In our running case system, the constructor of *Camera1* allocates memory space that holds 1280 x 960 pixel frames. When sensors provide frames with lower resolution and memory footprint, *Camera1* has reserved the same amount of memory (corresponding to 1280 x 960 pixel frames) from which it uses only a part, resulting in under utilization of the memory.

Besides the memory requirements, each component with GPU capability need a particular GPU computation resource (i.e., GPU thread) when processing images. For example, the *EdgeDetection* requires a number of 1280*960 GPU threads (i.e., a thread for each image pixel) to process the input image. When the camera sensors switch from 1280 x 960 to 640 x 480 pixel frames, *EdgeDetection* has reserved the same number of GPU threads corresponding to the high-resolution frame. This leads to a waste of GPU computation resources.

Another part of the underwater robot is the logger system that is composed of three software components, i.e., *CompressGrayscale*, *CompressRGB* and *Logger*. This part of the software architecture has a non-critical functionality. The *CompressGrayscale* and *CompressRGB* components have GPU capability, that is, their image compression functionality is executed on the GPU. The purpose of this non-critical part is to compress and record various information of the robot during the underwater journey. Due to the limited (CPU and GPU) memories, the logger system is triggered by a *clock* element to save the resulting frames, with a specific (low) time frequency. These frames are copied from the RAM to a flash memory by a specific service of the operating system. If the system has more available memory then the logger system is triggered with a faster time frequency. In this case, there will be an improvement in various system activities e.g., checking the (correct) functionality of the vision system by using a higher number of processed frames. Moreover, the logger system can benefit from extra memory by delivering other system information (e.g., energy usage and temperature) that improves the debugging activity of the robot.

IV. Proposed Solution and Proof-of-concept Implementation

A. Generic Solution

In order to improve the resource utilization of non-critical parts of the embedded systems, we introduce an automatic method that, during run-time, provides information on the additional available resources that can be used by the non-critical parts. The proposed method is presented in Figure 4, which consists of four blocks: (i) Critical System, (ii) Non-critical System, (iii) Monitor, and (iv) Evaluator. The Critical System and Non-critical System blocks represent the critical and non-critical functionalities in the system as discussed in Section I. The Monitor block periodically checks (e.g., every triggering execution) the resources usage of only the Critical System. Lastly, the Evaluator block, based on the information received from the Monitor, evaluates the available system resources and provides this information to the Non-critical System. The interactions among the different blocks in Figure 4 are identified by means of the arrows. Step (arrow) 1 from Figure 4 expresses the examination of the Critical System by the Monitor. During step 2, the Monitor sends the actual usage of the resources to the Evaluator. Based on the the received information, the Evaluator has the following two options:

- if the Critical system uses as much resources as its maximum (worst case) requirement, the Evaluator informs the Non-critical system to use its default allocated resources (step 3),
- if the Critical system uses less resources than its maximum requirement, the Evaluator computes the size of the unused resources and distributes it to the Non-critical system (step 4).

B. Realization

This subsection describes the realization details of our method using the vision system case study. The first part of the section introduces groundwork details on the functionality of the component model, while the second part presents the overall realization of our method.

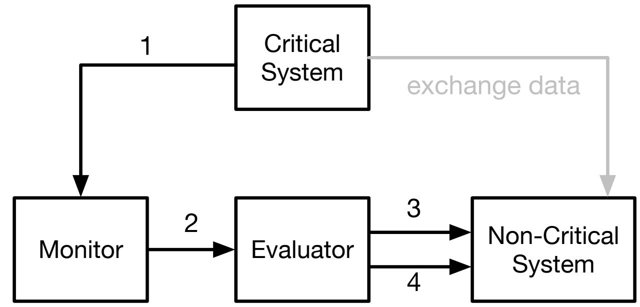


Figure 4. Overview design of the Evaluator method.

1) Component Model Functionality: Each component is characterized by a constructor and a destructor. The constructor is executed once, at the initialization of the system, and allocates as much memory as the component requires. The destructor, executed once when the system is properly switched off or reset, has the purpose to deallocate the memory. Figure 5 focuses on two connected software components from the vision system. In order to simplify the figure, we remove some of the (triggering) connections of the components. *Camera1* sends a frame to *MergeAndEnhance* component. Initially, the constructor of *Camera1* allocates memory space (on CPU memory space) to accommodate frames of maximum size (i.e., 1280 x 960 pixels). Similarly, the constructor of *MergeAndEnhance* allocates on GPU memory address, memory space for high-resolution frames (1280 x 960 pixels). Furthermore, each time when the component is executed it uses the same number of GPU threads (i.e., one thread per pixel) to process high-resolution frames. When the robot changes its mode (e.g., to save its energy) and its physical cameras send lower size frames (640 x 480 pixel frames), both *Camera1* and *MergeAndEnhance* use only a part of the memory allocated to them by their respective constructors. Moreover, *MergeAndEnhance* uses more GPU threads to process 640 x 480 pixel frames than it needs.

To send large data (i.e., larger than a scalar), components need to use pointers, as follows. The output port of *Camera1* is basically a *struct* that contains a pointer variable and two scalars, characteristics to 2D images. The port may cover other types of data, such as 3D images by including additional information, such as a third scalar. The pointer indicates to the memory address that it is at the beginning of the data to be transferred, and the two scalars (i.e., height and width) describe the size of the frame. In this way, *Camera1* passes the information (of the pointer and scalars) about the data to be transferred to the *MergeAndEnhance* component.

Figure 5 presents, in the lower part, an abstract overview of the hardware layer. In the CPU memory address section, a memory space is allocated for *Camera1* to hold 1280 x 960 pixel frames. This memory location is indicated by the

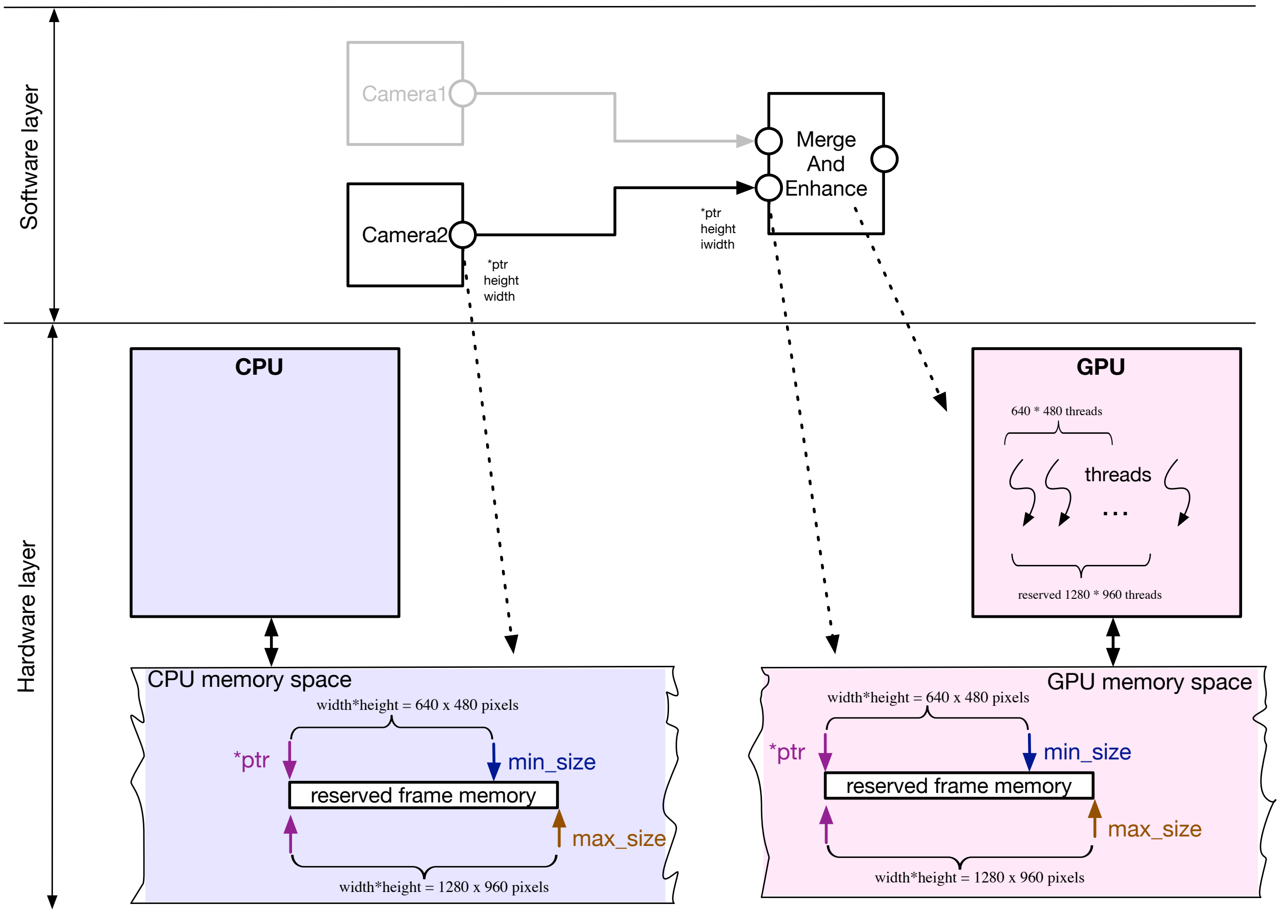


Figure 5. The requirements of two connected components.

pointer *ptr*. For frames of different resolution, *ptr* points to the same location. The difference is that for high resolution frames the size of the memory (i.e., *max_size*) is different (i.e., higher) than the size of the memory location corresponding to low-resolution frames (i.e., *min_size*). Similarly, the different sizes of memory allocation, in the GPU memory space section, corresponding to the high- and low-resolution frames in the case of *MergeAndEnhance* component are depicted in Figure 5. Furthermore, the figure presents, in an abstract way, the *MergeAndEnhance* requirements of GPU threads usage to process the frames of different sizes.

2) *Vision System Realization:* The vision system is composed of four parts and realized as follows.

a) The Critical System. The critical system contains the functionality that has the highest priority in the system. In our case, it produces and processes the frames, and takes decisions based on the findings. There are seven software components included in this part of the system as illustrated in Figure 6.

b) The Monitor. We realize the monitor as a service that is regularly performed by the operating system. The service checks the settings of the camera sensors and produces a value that corresponds to the frame sizes produced by the cameras, i.e., 1024 or 640.

c) The Evaluator. The evaluator is realized as a regular software component that receives its input information from the monitoring service. Because it decides the distribution of the resource memory utilized by the critical system, the priority of the *Evaluator* component is set to the highest level. Based on the value received from the monitor, the Evaluator component decides if the non-critical system can use more resources and produces the data that reflects this decision. For simplicity, the output result is a boolean variable; the output value 1 means that the non-critical system may use more resources than initially allocated, and 0 the opposite. The *Evaluator* component (i.e., its constructor, behavior function and destructor) is entirely automatically generated through our solution (see Section IV-C for details).

d) The non-critical system. The part of the system that handles the logging functionality represents the non-critical system. It has a lower priority than the critical system (and the *Evaluator* software component). It contains three software components, i.e., *CompressRGB*, *CompressGrayscale* and

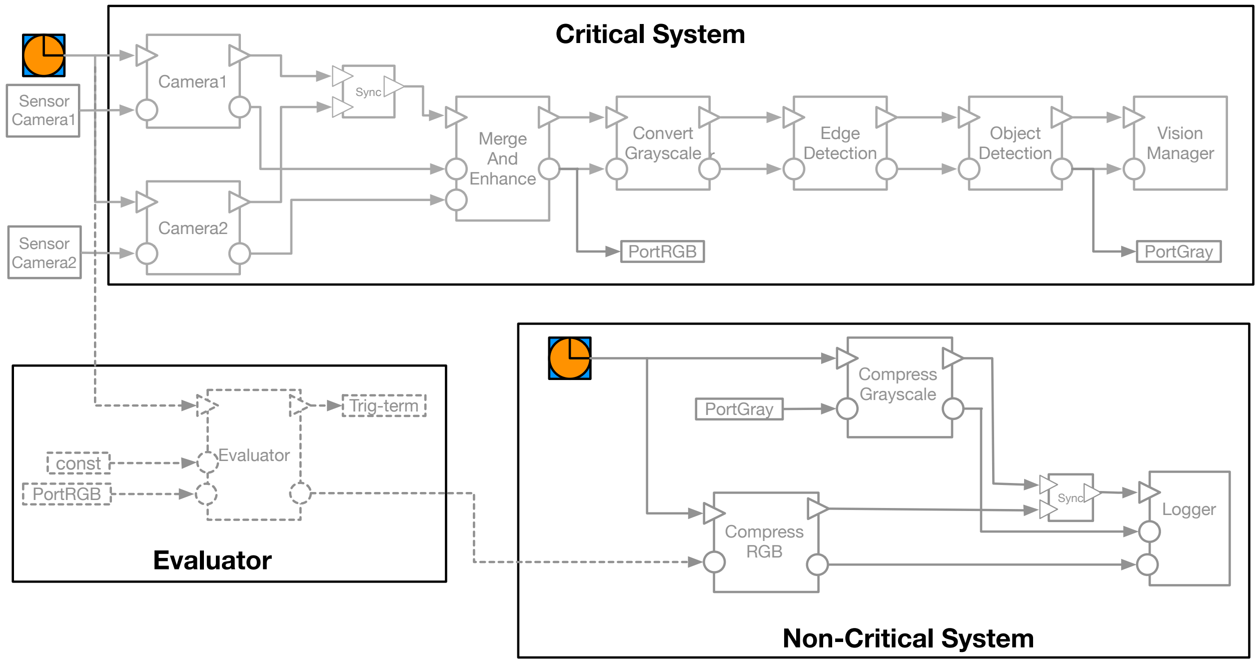


Figure 6. Realization of the Evaluator method applied on the vision system.

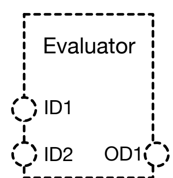
Logger that communicate with the *Evaluator* through an additional port. Based on the (boolean) input data received via the additional port, the non-critical components are triggered with a higher frequency rate, which leads to an improved system logger, with more compressed frames describing the underwater journey of the robot.

C. Implementation

The solution presented in this paper does not interfere with the development and execution of the critical system which is fully constructed by the developer. For the monitoring solution, we use a service provided by the OS, which is periodically executed in the background. Our solution realizes the Evaluator as a regular Rubus software component, in an automatic and transparent manner. The *Evaluator* component is generated, using the existing Rubus framework, with all its constituent parts, i.e., interface, constructor, behavior function and destructor. The interface of the component is realized as a header file, which is described in Figure 7. The interface contains two input data ports (i.e., *ID1* and *ID2*) and one output data port (i.e., *OD1*). The *ID1* port receives input data from the monitoring service, and *ID2* port receives the merged frame provided by the *MergeAndEngance* component.

The behavior function of the *Evaluator* component decides, based on the input data received from the monitoring service, to send or not the merged frame to be compressed. Figure 8 illustrates the functionality of the *Evaluator*. The output port is initialized with the image frame (line 2) received as input data, when the robot is on the low resolution mode (line 1), i.e., the Critical System does not use its entire allocated memory. Although the Evaluator functionality is simple and can be easily merged to the non-critical system, we opt for the separation-of-concerns principle, which is essential in the model- and component-based software development. Moreover, the evaluator functionality can be increased to adapt to more complex systems.

The non-critical system is mostly constructed by the developer, where our approach introduces some elements that are




```
typedef struct {
  unsigned_char *ptr;
  int width;
  int height;
}img_format;

/* input ports */
typedef struct {
  int *ID1;
  img_format *ID2;
}IP_SWC_iArgs;

/* output ports */
typedef struct {
  img_format OD1;
}OP_SWC_iArgs;

/* the interface declaration */
typedef struct {
  IP_SWC_iArgs IP;
  OP_SWC_iArgs *OP;
}SWC_Evaluator_iArgs;
```

Figure 7. The interface of the *Evaluator* component

```
if( mode == Low_RESOLUTION)
  IPA_OD1_Evaluator_ = {(void *)&
       SWC_Evaluator_iArgs->IP.ID2->ptr,
       SWC_Evaluator_iArgs->IP.ID2->width,
       SWC_Evaluator_iArgs->IP.ID2->height};

else IPA_OD1_Evaluator = {NULL, 0, 0};
```

Figure 8. Generated part of the Evaluator behavior function.

automatically generated. Initially, the non-critical system uses resources to process one frame (i.e., the grayscale frame); the constructors of *CompressGrayscale* and *Logger* components allocate memory for their functionality to compress and, respectively, log, the grayscale frame. In order to enforce a larger memory usage, the *CompressRGB* components needs to specifically allocate memory to hold the result from processing the merged frame. As the constructor is executed once at the system initialization stage, we automatically allocate memory inside the components' behavior function.

```
1  if(SWC_CompressRGB_iArgs->IP.ID1->ptr != NULL){
2         cl_mem frame_out = clCreateBuffer(context,
       CL_MEM_READ_WRITE,
       3*(SWC_CompressRGB_iArgs->IP.ID1->width) *
       (SWC_CompressRGB_iArgs->IP.ID1->height) *
       sizeof(unsigned char), NULL, NULL);
3  }
4  else {
5     IPA_OD1_CompressRGB = { NULL, 0, 0 };
6     return 0;
7  }
8
9  /* initialize parameters */
10   clSetKernelArg(kernel, 0, sizeof(cl_mem), (void
        *)&SWC_CompressRGB_iArgs->IP.ID1->ptr);
11   clSetKernelArg(kernel, 1, sizeof(int), (void *)
       &SWC_CompressRGB_iArgs->IP.ID1->width);
12   clSetKernelArg(kernel, 2, sizeof(int),(void *)&
       SWC_CompressRGB_iArgs->IP.ID1->height);
13   clSetKernelArg(kernel, 4, sizeof(cl_mem), (void
        *)&frame_out);
14
15 /* execute functionality on the input frame */
16   clEnqueueNDRangeKernel(command_queue, kernel,
       2, NULL, global_size, local_size, 0, NULL,
       NULL);
```

Figure 9. Part of the behavior function of *CompressRGB* component.

Figure 9 illustrates a section of the behavior function of the *CompressRGB* component. For the component realization, we introduce rules to generate the code from line 1 to 10. The generated code checks the input frame sent from the Evaluator (line 1). In the case that the frame exists (i.e., is not NULL), memory is specifically allocated to hold the result from processing the input merged frame (line 2). In the opposite case (i.e., the frame is NULL), the output data port is initialized with an empty frame (line 5), and the behavior function is exits (line 6). The rest of the function is defined by the component developer and is specific to the GPU functionality implemented using the OpenCL [22] syntax. Parameters that correspond to the frame specifications are set in lines 10-13, and the functionality of the component is triggered to be executed on the GPU (in line 16).

V. Evaluation

This section focuses on the evaluation of the overhead (the effect of additional elements) incurred due to the proposed solution. There are two parts that influence the overall overhead, i.e., the memory footprint and the execution time.

The memory footprint refers to the generated *Evaluator* component and the generated part of the behavior function of *CompressRGB* component (see Figure 9). The Evaluator component consists of a constructor, behavior function, and a destructor. Moreover, it has specification of its interface (i.e., ports) in a separate header file. The memory footprint of all of its code takes approximately 14 KB. We need to also add the memory size occupied by the generated parts of the *CompressRGB* component, which result in a total of 15 KB. We consider that the memory footprint overhead resulted from our approach is manageable for an embedded systems with GPUs, compared to traditional (CPU-based) embedded systems. The CPU-GPU embedded systems are characterized by a reasonable high amount of memory (i.e., order of tens of Megabyte) due to the computation power that requires high memory specifications.

TABLE I. The memory requirement of the Critical System

Component name	Memory requirement (kB)	
	Low-mode*	High-mode**
Camera1	165	307
Camera2	165	307
MergeAndEnhance	298	536
ConvertGrayscale	197	356
EdgeDetection	86	154
ObjectDetection	25	45
VisionManager	14	26
Total	950	1731

Low-mode* - camera images have 640 * 480 pixels.
High-mode** - camera images have 1280 * 960 pixels.

Besides the memory footprint, the Critical System also has a memory requirement regarding the data to be processed. Table I presents the requirement of each component of the Critical System, in the two modes of the robot. For example, *MergeAndEnhance* component, during the low-resolution mode (i.e., camera frames with 640 * 480 pixels), uses 298 kB of memory, while during the high-resolution mode, uses 536 kB of memory. During the system initialization, the Critical System is allocated, to process data, with an amount of 1731 kB of memory. When the robot switches to the low-resolution mode, our method provides the unused 781 kB of memory of the Critical System to the Non-critical System.

Regarding the execution time, the generated *Evaluator* component may negatively affect the execution time of the critical system. In this regard, we conducted an experiment to compare the performance with and without our approach. The system on which we executed the experiments contains an embedded board AMD Accelerated Processing Unit with a Kabini architecture (i.e., CPU-GPU SoC). We used two input images, i.e., one with $640 * 480$ pixels and the other with $1280 * 960$ pixels. For each set of images, we executed two cases, one with and the other without our solution. Each case was executed 1000 times and we calculated its average execution time.

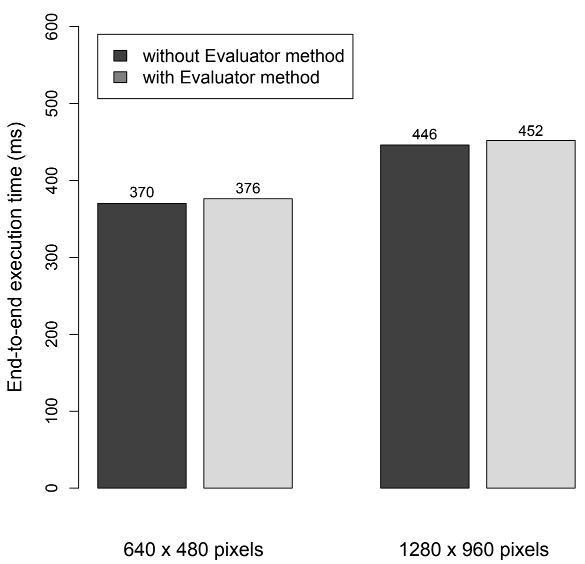


Figure 10. Usage of the Evaluator method in the vision system execution.

The results of the experiments are shown in Figure 10. A slight increase (1.3 to 1.6%) in the execution time can be observed when our solution is applied. The results indicate that the performance of the non-critical part of these systems can be significantly improved with our method at the very small execution time overhead.

Furthermore, both versions of the vision system produced the same outputs (i.e., frames). Compared to the original version, in the vision system version implemented using our approach, besides the frames produced by the *ObjectDetection* and *CompressGrayscale* components, the *CompressRGB* component also produced frames. With more data to analyze, the logger functionality of the system that is implemented using our solution showed improvement over the original version.

Besides the memory aspect evaluated in this section, our solution also deals with the GPU computation resources. The released number of GPU threads used by the critical system may be employed by the other part of the system that may use GPU simultaneously with the critical system. As the GPU allows, depending on its specifications[1], concurrent execution of several components, and by freeing an amount of GPU threads, the overall system performance may be further improved by distributing the available resources to other parts of the system that have GPU requirement.

VI. Related Work

There exist different methods to increase the memory utilization, which are presented in various surveys [23]. We mention a solution to reduce the actual allocated space for temporary arrays by using a mapping of different array parts into the same physical memory [24]. Another method proposes scratch pad memories to reduce the power consumption and improve performance [25]. These solutions are applicable at a very low level of abstraction and are not suitable to be merged with our approach, which is applicable at the implementation abstraction level where the software architecture of the application is modeled.

Regarding monitors, many works utilize them for different purposes, such as data-flow monitoring solutions to simulate large CPU-GPU systems [26], and GPU monitors for balancing the bandwidth usage [27]. An interesting work conducted by Haban et al. [28] introduces software monitors to help scheduling activities. The authors described the low overhead of the monitoring solutions, which degrade the CPU performance with less than 0.1%. In our work, we use the same type of monitors analyzed by Haban (i.e., software monitors) that have a low impact over the system performance.

In the context of developing and extending component models for embedded systems, Campeanu et al. [29] [30] introduce a solution to facilitate the component-based software development of systems with GPUs. The solution is implemented as an extension of the Rubus component model, and introduces *platform-agnostic components* with GPU capability and *adapters*. The platform-agnostic components are components that can be re-utilized on various types of platforms with GPUs, without any manual change. The adapter is a concept that facilitates the communication between components that are executed by different processing units. For example, when a CPU-executed component communicates with a GPU-executed platform-agnostic component, the adapter connects the two components and transfers, in an automatic way, the data between the CPU and GPU memory addresses. Although these two concepts were not introduced in this work, in order to mitigate the complexity of the solution, our introduced method builds upon these concepts. For example, an adapter is the artifact that transfers the data from *Camera2* to *MergeAndEnhance*, between the CPU and GPU memory spaces (see Figure 5).

Model-driven development is another paradigm adopted in the development of embedded systems. In this context, we mention the work of Rodrigues et al. that facilitate modeling

[1]e.g., an NVIDIA GPU with the Pascal architecture and compute capability 6.1 can concurrently execute up to 32 activities

of embedded systems with GPUs [31]. Due to the fact that the work has been developed in 2013, it is limited in covering the recent platform advancements. In the same context, we mention other works such as the MARTE-based framework proposed by Gamatie et al. which automatically allows generation of code for heterogeneous platforms [32].

There exists on the market various models that facilitate the development of applications with GPU capabilities. We mention the CUDA model [33], that is developed by NVIDIA vendor to specifically handle their own GPUs. CTM is another model developed by AMD to address ATI AMD GPUs [34]. OpenCL [22] is a general framework that is supported by different types of processing units (i.e., CPU, GPU, FPGA) produced by various vendors (e.g., NVIDIA, AMD, Intel). Although, in this work, we do not specifically address the functionality of the components with GPU capability such as *MergeAndEnhance* and *CompressGrayscale*, OpenCL framework was used to develop the GPU functionalities of the components used for the vision system used in the evaluation part.

VII. Conclusion

Modern embedded systems deal with huge amount of data that is originated from their interaction with the environment. GPUs have emerged as a feasible option, from the performance perspective, for processing the huge data inputs. However, with GPU-based solutions, the resource utilization remains high, which is an important aspect when dealing with resource-constrained embedded systems. In this paper, we have presented a method that improves the resource utilization for non-critical parts of CPU-GPU-based embedded systems. Whenever the critical part of the system does not fully utilize its resource requirements (i.e., memory and GPU threads) due to various reasons, such as reducing energy consumption, the presented method distributes the unused resources to the non-critical parts of the system. With the availability of more resources, the performance of the non-critical parts of the system is improved without effecting the performance of the critical parts. As a result, the performance of the overall embedded system is improved.

As a proof of concept, we have realized the method in a state-of-the-practice component model, namely the Rubus Component Model. We have also demonstrated the usability of the method using the underwater robot case study. The evaluation results indicate that the proposed method can significantly improve the performance of non-critical parts of CPU-GPU-based embedded systems at the cost of very small execution time overhead of approximately 1.5%.

An interesting future work is to extend the presented method to support the heterogeneous system architectures that include an FPGA, beside the GPU. In a system with multiple accelerators, our introduced Evaluator should be extended to decide, based on the available resources, on where the Non-critical System should execute its functionality in order to improve the overall system performance. Another future work is to model the proposed solution as a self-adaptive system that is built using the feedback control loops [35].

Acknowledgments

The work in this paper is supported by the Swedish Foundation for Strategic Research (SSF) and the Swedish Knowledge Foundation (KKS) through the projects RALF3 (IIS11-0060) and PreView respectively. We thank our industrial partners Arcticus Systems, Volvo CE and BAE Systems Hägglunds.

References

[1] G. Campeanu and S. Mubeen, "Improving run-time memory utilization of component-based embedded systems with non-critical functionality," ICSEA, 2017, pp. 139–144.

[2] Google. Waymo - Google Self-Driving Car Project. https://waymo.com/. Accessed: 2018-02-15.

[3] NVIDIA, "NVIDIA Jetson TK1," http://www.nvidia.com/object/jetson-tk1-embedded-dev-kit.html, accessed: 2018-02-15.

[4] AMD, "Embedded R-Series Family of Processors," http://www.amd.com/en-us/products/embedded/processors/r-series, accessed: 2018-02-15.

[5] Samsung, "Exynos 8 Octa," http://www.samsung.com/semiconductor/minisite/Exynos/w/solution/mod_ap/8890/, accessed: 2018-02-15.

[6] See Fast Technologies. High Speed Camera ProcImage500-Eagle. http://www.seefastechnologies.com/procimage-eng1-pi500-eagle.html. Accessed: 2018-02-15.

[7] J. D. Owens, M. Houston, D. Luebke, S. Green, J. E. Stone, and J. C. Phillips, "GPU computing," Proceedings of the IEEE, vol. 96, no. 5, 2008, pp. 879–899.

[8] S. A. Manavski, "CUDA compatible GPU as an efficient hardware accelerator for AES cryptography," 2007, pp. 65–68.

[9] T. Preis, P. Virnau, W. Paul, and J. J. Schneider, "GPU accelerated Monte Carlo simulation of the 2D and 3D Ising model," Journal of Computational Physics, vol. 228, no. 12, 2009, pp. 4468 – 4477.

[10] J. E. Stone, J. C. Phillips, P. L. Freddolino, D. J. Hardy, L. G. Trabuco, and K. Schulten, "Accelerating molecular modeling applications with graphics processors," Journal of computational chemistry, vol. 28, no. 16, 2007, pp. 2618–2640.

[11] T. A. Henzinger and J. Sifakis, "The embedded systems design challenge," in International Symposium on Formal Methods. Springer, 2006, pp. 1–15.

[12] I. Crnkovic and M. P. H. Larsson, Building reliable component-based software systems. Artech House, 2002.

[13] D. Brugali and P. Scandurra, "Component-based robotic engineering (part i)[tutorial]," IEEE Robotics & Automation Magazine, vol. 16, no. 4, 2009, pp. 84–96.

[14] D. Box, Essential COM, 1st ed. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1997.

[15] R. Monson-Haefel, Enterprise JavaBeans. Sebastopol, CA, USA: O'Reilly & Associates, Inc., 1999.

[16] I. Crnkovic, S. Sentilles, A. Vulgarakis, and M. R. Chaudron, "A classification framework for software component models," IEEE Transactions on Software Engineering, vol. 37, no. 5, 2011, pp. 593–615.

[17] "AUTOSAR - Technical Overview," http://www.autosar.org, accessed: 2018-02-15.

[18] K. Hanninen, J. Maki-Turja, M. Nolin, M. Lindberg, J. Lundback, and K.-L. Lundback, "The rubus component model for resource constrained real-time systems," in Industrial Embedded Systems, 2008. SIES 2008. International Symposium on. IEEE, 2008, pp. 177–183.

[19] I. Application, "Implementation of IEC 61131-3," Geneva: IEC, 1995.

[20] S. Sentilles, A. Vulgarakis, T. Bures, J. Carlson, and I. Crnkovic, "A Component Model for Control-Intensive Distributed Embedded Systems," in 11th International Symposium on Component Based Software Engineering (CBSE), vol. 8. Springer, October 2008, pp. 310–317.

[21] C. Ahlberg, L. Asplund, G. Campeanu, F. Ciccozzi, F. Ekstrand, M. Ekstrom, J. Feljan, A. Gustavsson, S. Sentilles, I. Svogor et al., "The Black Pearl: An autonomous underwater vehicle," 2013.

[22] KHRONOS Group, "The OpenCL specifications," https://www.khronos.org/developers/reference-cards/, accessed: 2018-02-15.

[23] P. R. Panda, F. Catthoor, N. D. Dutt, K. Danckaert, E. Brockmeyer, C. Kulkarni, A. Vandercappelle, and P. G. Kjeldsberg, "Data and memory optimization techniques for embedded systems," ACM Transactions on Design Automation of Electronic Systems (TODAES), vol. 6, no. 2, 2001, pp. 149–206.

[24] M. A. Miranda, F. V. Catthoor, M. Janssen, and H. J. De Man, "High-level address optimization and synthesis techniques for data-transfer-intensive applications," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 6, no. 4, 1998, pp. 677–686.

[25] R. Banakar, S. Steinke, B.-S. Lee, M. Balakrishnan, and P. Marwedel, "Scratchpad memory: design alternative for cache on-chip memory in embedded systems," in Proceedings of the tenth international symposium on Hardware/software codesign. ACM, 2002, pp. 73–78.

[26] B. R. Bilel, N. Navid, and M. S. M. Bouksiaa, "Hybrid CPU-GPU distributed framework for large scale mobile networks simulation," in Proceedings of the 2012 IEEE/ACM 16th International Symposium on Distributed Simulation and Real Time Applications. IEEE Computer Society, 2012, pp. 44–53.

[27] M. K. Jeong, M. Erez, C. Sudanthi, and N. Paver, "A QoS-aware memory controller for dynamically balancing GPU and CPU bandwidth use in an MPSoC," in Proceedings of the 49th Annual Design Automation Conference. ACM, 2012, pp. 850–855.

[28] D. Haban and K. G. Shin, "Application of real-time monitoring to scheduling tasks with random execution times," IEEE Transactions on software engineering, vol. 16, no. 12, 1990, pp. 1374–1389.

[29] G. Campeanu, J. Carlson, and S. Sentilles, "Developing CPU-GPU embedded systems using platform-agnostic components," in Software Engineering and Advanced Applications (SEAA), 2017 43rd Euromicro Conference on. IEEE, 2017, pp. 176–180.

[30] G. Campeanu, J. Carlson, S. Sentilles, and S. Mubeen, "Extending the Rubus component model with GPU-aware components," in Component-Based Software Engineering (CBSE), 2016 19th International ACM SIGSOFT Symposium on. IEEE, 2016, pp. 59–68.

[31] A. W. O. Rodrigues, F. Guyomarc'H, and J. L. Dekeyser, "An MDE approach for automatic code generation from UML/MARTE to OpenCL," Computing in Science & Engineering, vol. 15, no. 1, 2013, pp. 46–55.

[32] A. Gamatie, S. Le Beux, E. Piel, R. Ben Atitallah, A. Etien, P. Marquet, and J.-L. Dekeyser, "A model-driven design framework for massively parallel embedded systems," ACM Trans. Embed. Comput. Syst., 2011, pp. 39:1–39:36.

[33] "NVIDIA CUDA C programming guide," https://docs.nvidia.com/cuda/cuda-c-programming-guide/, accessed: 2018-02-15.

[34] J. Hensley, "AMD CTM overview," in ACM SIGGRAPH 2007 courses. ACM, 2007, p. 7.

[35] J. O. Kephart and D. M. Chess, "The vision of autonomic computing," Computer, vol. 36, no. 1, Jan 2003, pp. 41–50.

Using Qualitative System Dynamics in the Development of an Agile Teamwork Productivity Model

Israt Fatema
Institute of Information Technology
University of Dhaka
Dhaka, Bangladesh
email: mph001@iit.du.ac.bd

Kazi Sakib
Institute of Information Technology
University of Dhaka
Dhaka, Bangladesh
email: sakib@iit.du.ac.bd

***Abstract*— Improvement in agile software development will not be achieved without considering that there is a large number of factors affecting agile teamwork productivity. The objective of this study is to explore what factors influence agile teamwork productivity, and how these factors interacted. This is achieved through a two-phase approach. The first phase involves reviewing relevant literature, performing a set of in-depth interviews with agile team members and conducting a survey to identify productivity factors. The survey has been administered to 60 respondents from 18 agile software companies in Bangladesh. The second phase involves the construction of qualitative system dynamics model (causal loop diagrams) of agile teamwork productivity with the findings from the first phase to analyze the productivity influence factors. The findings from the first phase reveal the most perceived factors are motivation, team effectiveness and team management. Social hierarchy culture influences self- managed agile team from correct implementation of agile practice. Although, the most followed organizational structure is horizontal, Scrum is leading agile practice among the participating companies. Lack of management support is found to be the most mentioned reason for any failed agile project. The resulting qualitative model is expected to provide more insight into the agile teamwork dynamics and establish a basis for a further quantitative modelling.**

Keywords-agile teamwork productivity; influence factors; qualitative system dynamics; social hierarchy culture; team effectiveness; causal loop diagram.

I. INTRODUCTION

The objective of any software company is to be efficient and productive by being cost effective and time optimum. Agile process places more emphasis on people factors in the project. Therefore, agile teamwork productivity is one of the most important aspects in achieving project success at different stages of a project. Improvement in agile software development will not be achieved without considering that there is a large number of factors affecting agile teamwork productivity. It is important to identify the most influential ones among the factors and develop policies to manage them. A better understanding of the factors influencing agile teamwork productivity can enable team management to determine where efforts are to be directed in order to improve productivity.

This paper explores which factors influence agile teamwork productivity and how these factors impact productivity of agile team. It thereby revises and extends previous contribution [1], by an in-depth discussion of the complex inter-related structure of different factors. A System Dynamics (SD) based approach has been used to model agile teamwork productivity influence factors. SD introduced by Jay Forrester of the MIT in the 1960s as a modelling and simulation methodology for studying complex system [2].

There are different factors that affect the productivity of traditional software development teams as pointed out by previous researchers [3][4]. Although agile approaches have become popular with a wide variety of organizations, there is insufficient evidence on the effect of agile productivity factors [5]. However, agile teamwork productivity is a function of various controllable and uncontrollable factors [6]. Software productivity across many projects, culture and practice varies extensively even if the same type of software is developed [7]. The software industry is also different from country to country as are the resource availability, the laws, which govern it and the developer's cost [8]. In addition, actual productivity measurement becomes more difficult when agile software developers perform knowledge-related tasks (e.g., creating, storing, sorting, retrieving, applying and acquiring knowledge) where the product is usually intangible, rarely has single way of doing it, and it is difficult to quantify [6]. Since knowledge is complex and hard to evaluate, it is difficult to interpret the productivity of the agile team member's simply by Source Line Of Code (SLOC) or function points produced per unit of time/cost [7]. Measuring the same code gives different results with different code counters since there is no universal standard for Line Of Code (LOC) [9]. Story points, used in agile software development, are very subjective and metrics based on story points cannot be used to compare between teams, units or organizations [10].

It would be helpful if the productivity influence factors can be controlled by the Project Manager (PM) when establishing and managing an agile project. "You cannot control what you cannot measure" [11]. Nevertheless, it is difficult to measure agile teams' productivity [10]. In view of the fact that agile software development empowers self-managing teams instead of forming traditional project management. Consequently, project has less control on the management level. To overcome this limitation and domination project activities, a clear list of influences on productivity in agile software development is needed. Agile team members also should learn to interpret and direct productivity factors regularly as they are self-managed. The researchers have highlighted the value of team learning to help organization achieving team effectiveness, better ways to resolve problem and improve productivity.

In an earlier work [1], identification of agile teamwork productivity influence factors was conducted on seventeen software companies in Bangladesh. The study showed the main perceived factors impacting on agile team member's productivity were team effectiveness, team management, motivation and customer satisfaction. Lack of management support was found to be the most mentioned reason for any failed agile project.

This extension of the previous work added more software companies and survey respondents from agile teams to provide a better understanding of agile teamwork productivity influence factors. The qualitative model of agile teamwork productivity is developed from these sources using commanding cause and effect feedback loop.

Moreover, if the cause-effect relationships between the factors can be clarified and quantified, quantitative models or formulations could be established. Therefore, the future contribution of this research shall provide a strategic (quantitative) model that tells the PM in advance about the degree of impact these factors will have on teamwork. Using the proposed model, PM may identify the origin of a decrease in productivity. As a result, the agile teamwork productivity may be improved by implementing of management strategies.

The rest of this paper is organized as follows. Section II includes a literature review, Section III presents the research method and design. Section IV describes the survey results and Section V explores the structure of the qualitative SD model. Section VI presents causal loop diagram validation and Section VII describes some limitations of this work. Finally, Section VIII contains the conclusion and future work.

II. LITERATURE REVIEW

Agile adoption is growing within organizations for accelerating software delivery and productivity, it is essential to discern whether the factors influencing productivity remain the same in all context [6].

Dingsoyr et al. described agile software development as a sociotechnical system comprised of human (socio) and technical entities, the culture of the society in which the system works is crucial [12]. Technological interventions do not increase sociotechnical system effectiveness if they are not supported by social (self-managing team and group) components of the system. Thus, recent focus on agile software development has increased interest in analysing self-managing agile team and how to effectively make team productive [12].

According to the Agile Manifesto, it focuses on individuals and interactions between people (teamwork) over processes and tools [13]. Therefore, agile software development is influenced by the underlying values and background of the people involved with development process. These personal characteristics of the people are very much influenced by their local tradition [14]. A survey study by Verner et al. reveals that teamwork productivity factors differ across countries and, culture influences teamwork's decision-making process, problem solving approach, social interaction, satisfaction and expectation [15].

There are several studies that attempted to assess the impact of some of the influencing factors on agile teamwork productivity. Besides, agile surveys have been conducted mostly on development process and overall view of agile practices [16][17][18]. However, these surveys do not elaborate much on productivity factors, and do not consider how they are related. Only Melo et al. analysed the major factors influencing agile teamwork productivity using the team's perception as one potential dimension to understand their overall productivity [6][18]. Through perceptions, they found that team management is the most influencing factor on agile team productivity.

SD technique has been applied in software engineering fields for modelling purposes, which is important for the organization and the project. SD is well suited for studying complex systems where unknown attributes of system properties are less visible [2]. There are few researches that attempted to evaluate the impact of some of the influencing factors on productivity separately using SD [5][19][20][21]. However, the complex inter-related structure of all the major factors influencing the teamwork productivity was not considered by the previous works. Abdel-Hamid and Madnick attempted to integrate system dynamics modelling and project dynamics insights with traditional software development processes [19]. In [21], Abdel-Ahmed investigated the effect of various management policies on development cycle time, quality and effort. However, his works adopt the waterfall method, which limits their applicability in recent software project and more importantly, does not focus on the agile principles.

In another research, Rodrigues proposed methods by which system dynamics modelling can be integrated with principles of project management [22]. In [23], the authors discussed whether agile project will fit within the common system dynamics project management structures or it has a unique structure. An analysis of factors that impact on productivity during agile web development and maintenance phases was conducted by Kong et al. [5]. However, it does not explicitly show the interrelations of different variables that influence the effectiveness of teamwork. Cao et al. created an integrative system dynamics model of agile software development for investigating refactoring and its impact [24]. The authors investigated the dynamics of agile software development and the impact of agile practices on cycle time and customer satisfaction using SD [24]. Modelers have also investigated schedule pressure effects on the dynamics of iterative development cycles [25].

In addition, evaluation of individual productivity may not affect the productivity of other team members [18]. These ideas provide a motivation to study teams' productivity, not individuals. A number of studies exist on teamwork in agile software development on a range of topics relevant to composition of team [26], task-effective norms in teams [27], team member's motivation [28], and the importance of a team vision. Yet others have focused on how team uses daily stand-up meetings to communicate [26], how team makes decisions [29] and how to achieve self-management [26].

Another stream of research has focused on team performance in agile software development to analyse the teamwork. Team performance refers to evaluation of the results of the teamwork. Moe et al. used a team performance model to explain teamwork in a project adopting Scrum: The Dickenson McIntyre model [12]. Melo et al. used the 'Input Process Output' model to identify team productivity factors in a multiple case study [6]. Boehm reported in his productivity estimation model, Constructive Cost Model (COCOMO), that productivity of a software development project is mostly affected by the development team and their team management [30]. Scacchi also identified that poorly managed or organized project's productivity was mostly lower than those projects, which were well managed [26].

Throughout the literature review, it has been observed that there is a lack of well-established dynamic theory about agile teamwork. This study seeks to fill this gap by developing an integrated model, which represents the inter-related structure of productivity influence factors and how they impact (positively or negatively) agile teamwork's productivity. In order to do so, this study applies a system dynamics approach, which can study complex system by exploring underlying associations and connections between the components of a system that normally are not discovered by the input-output-process type of models used in organizational studies. Focusing on people and teamwork aspects of agile team, this paper makes use of two team effectiveness models for better analysis of agile software development teamwork productivity. Two models, the Salas and the Dickenson McIntyre models are used, which focus on team effectiveness, and mainly on internal aspects of the team [12][23][26]

III. RESEARCH METHODOLOGY

The shortcoming of previous research studies lies in their not considering the complex inter-related structure and causal relationships of different factors (hard and soft) affecting the agile teamwork productivity. Thus, this study aims to develop a productivity model to analyse the interactions among the main factors of agile software development teamwork productivity. The research question in this study is therefore "*which factors do have an influence on agile teamwork productivity, and in what way (positively or negatively)?*" To answer this question, a qualitative system dynamics approach has been used to capture the interactions and causal relationship between the influencing factors. SD is a simulation methodology enables to model complex system considering all the influencing factors [31]. There are many modelling techniques developed and used so far, according to the modelling objective and perspective. However, system dynamics modeling chosen for this research because it provides a systematic method for description, exploration and inspection about the dynamic behavior of complex systems [32]. SD methodology has been applied by many researchers [31][33][34][35] for studying and managing complex feedback system, where feedback is understood as a closed sequence of causal relationships. The concept of a feedback loop reveals that any actor in a system will eventually be affected by its own action.

Figure 1 presents a flowchart of the main stages and activities involved in creating and validating the qualitative (Causal Loop Diagrams CLD) model of the agile teamwork productivity. As it can be seen in this flowchart, model building in SD begins with identifying and listing (Step 1) those factors that have a major influence on the output.

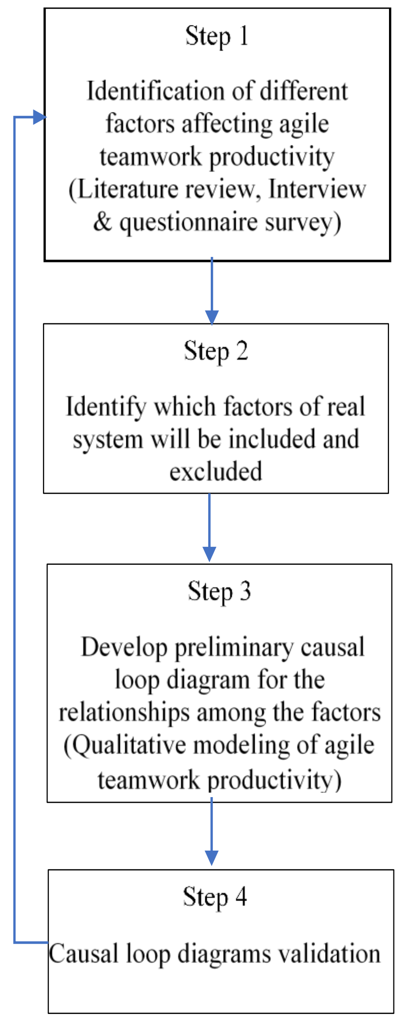


Figure 1. The development stages of the proposed qualitative SD agile teamwork productivity model

There are several methods established to identify those influences such as observation, discussion, interviews, existing data and survey [2]. This study chose to develop a questionnaire and conduct a semi-structured interview with responses based on the perception and observation of agile practitioners in Bangladeshi software companies. Survey research has been used because questionnaires are reasonable, completed within limited time and data can be interpreted through simple descriptive statistics [18]. Interviews were semi-structured in which interviewer did not strictly follow a specific set of questions. It allows more open-ended questions for a discussion with the interviewee [14]. Thus, semi-structured interview has been adopted to explore the views of the factors impacting agile productivity in the team's perception and how they influenced. Team members are central to the software development and consequently, they directly influence the team's productivity [36]. Therefore, team member perceptions used as a survey response in this study to analyse teamwork productivity influence factors.

Step 2 involves selection of factors to be included in the CLD. Then the qualitative model of teamwork productivity is constructed (Step 3) using causal relationships (cause and effect feedback loops) among identified factors affecting teamwork productivity. Model validation is conducted (Step 4) to ensure that the model adequately represents the real system.

A. Identification of different factors affecting agile teamwork productivity

Data collection: There are three important objectives of collecting information; to determine what factors affected productivity of agile team members, to determine how these factors impacting project productivity in the team's perception and to determine the significance of the factors.

1) Literature review: Keywords such as "productivity", "agile productivity influence factors", "system dynamics" and "agile teamwork" were used to search for related work in digital libraries. Significant findings from related work were not only helped in identifying some factors but also helped in the determination of the impact the factors have (positive or negative) on other variables in the project. The estimation of this impact would be vital in the calibration of the SD model.

2) Interview: Primarily, to collect qualitative data, a set of semi-structured interviews and face-to-face discussions were conducted with twelve key project members from four software companies in Bangladesh. All of the respondents had experience in agile software development methods, such as XP and Scrum. The roles of the respondents included project managers, scrum masters, developers and project owners. The semi-structured interviews mainly focused on their working team, their team productivity influence factors and experience of introducing agile practices in Bangladesh.

3) Questionnaire/survey: In an attempt to identify the perceived influencing factors and their impact on agile team members, data was collected with the help of online survey.

a) Questionnaire design: Using the factors identified in literature review and interview, a questionnaire consisting of 17 questions was developed [37]. Most of the questions were based on a previous global survey on agile methods conducted by [16] and country specific survey on agile productivity factors [18]. The questionnaire was structured in 4 parts. The first section was on demographic data, information about the project and organization. The respondents' details included their experience with agile methods, current position, current working project and status, working team size and organization name. The organizational profile included details about its' main activity, structure, size, mostly followed agile method, agile practice adopted and mostly used programming language. The second section was on perception of success/failed project and criterion for measuring/perceiving productivity. The third section was a set of 35 productivity influence factors. The last section was taken for any additional comments in order to allow the respondents to express their opinion more freely. To measure significance of agile teamwork productivity influence factors, the respondents were asked to indicate the strength (high, medium or low) for each factors that they perceived influenced their productivity.

b) Questionnaire administration and selection of respondents: The questionnaire was emailed to a total of 25 software companies in Bangladesh, requested for distribution within the organization through Human Resources departments. The company selection criteria for this preliminary study were: companies using agile methods for at least 1 year, developing software for both offshore and local market, and top listed software companies in Bangladesh [38]. Survey notifications were also sent to members of Agile-related group (Agile Bangladesh) with announcements on the Facebook group. 60 responses from 18 companies responded to the questionnaire. In the online survey, respondents were requested to fill up the questionnaire based on ongoing project or they had completed recently (regardless whether the project outcome was positive or negative). Data were collected throughout a period of eight months in 2017 (January-August). In order to ensure the quality of data, team members were all self-selected by their organization based on their work roles as members of existing agile teams. Therefore, respondents responded to survey questionnaires were already aware of agile team environment and mostly experienced. The filled-in questionnaires were then analysed to identify factors, which have major influences on agile teamwork productivity. Currently, more software companies are being requested to participate in this survey, as the plan is to collect more than 150 responses from different agile teams.

4) Author's assumption: Where necessary, author's assumptions are used in the development of the model. Such assumptions will be permitted and perhaps, moderated by experienced agile practitioners via interviews and questionnaire.

B. Selection of factors for inclusion in the model

Data analysis: Factors affecting agile teamwork productivity are rarely independent of the others, but a set of factors interacting with each other to build the final result [19]. The important factors identified in literature and interviews were taken as a starting point for the system approach in this research. In total, 35 factors were chosen for preliminary analysis. In order to create a system model to analyse the teamwork productivity, it is required to determine the importance of the individual factor, their correlation with one another and their effects on productivity itself. The agile team members were asked to fill in the questionnaire to indicate the strength (high, medium or low) of the factors that they perceived influenced their productivity [35].

The procedure followed to extract the agile team member's perception of the influence factors affecting their productivity can be summarized in the following steps:

1. Convert the qualitative scale to a quantitative one. The qualitative scale of high, medium or low was converted to a number scale of 3, 2, and 1, respectively.
2. Find the total score of each factor for frequency analysis. Then, the arithmetic mean of the total counts was calculated to eliminate the factors below the average (Table I) mean 2.26.
3. Cronbach's Alpha (α) coefficient for internal consistency reliability was calculated for the identified factors [39]. Cronbach alpha (α) is widely used as an estimator for reliability tests [39]. In a good solution for indicating high internal construct validity, Cronbach alpha ranges between zero and one - the larger the value, the more stable the factors. Generally, the value of 0.70 is accepted as the minimum desired value of reliability [39]. In this study, the 35 factors were tested for internal consistency, using the 60 respondent's data. The results, shown in Table I had values ranging from 0.877 to o.887, all of which were considered acceptable (Cronbach's alpha higher than 0.70) and Cronbach's alpha for 35 factors was .885
4. From step 2, twenty factors (Table I, highlighted) were selected as the most influential ones (above average mean)

IV. SURVEY RESULTS

This study used reliable survey instruments, that can be helpful for comparing new results with the previous studied results [16][18]. However, there is no data available on the state of agile development in Bangladeshi software companies to interpret this study sample representativeness. Interestingly, this study also found some similarities between Bangladesh and worldwide surveys [16][18].

This section presents a summary of the results found in this research. Characteristics of the sample software companies and respondents can be found in Figure 2 to Figure 11. As can be seen from Figure 2 – Figure 4 summarizes the respondents profile. The results show that 35% of the respondents cover the range 2-5 years of experience of using agile methods.

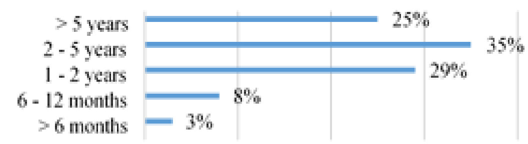


Figure 2. Agile practices experience

The respondents were working in various positions in their organizations, ensuring diversity in the survey. Figure 3 and Figure 4 present the team role of the respondents in their organizations and the respondent's main team assignment.

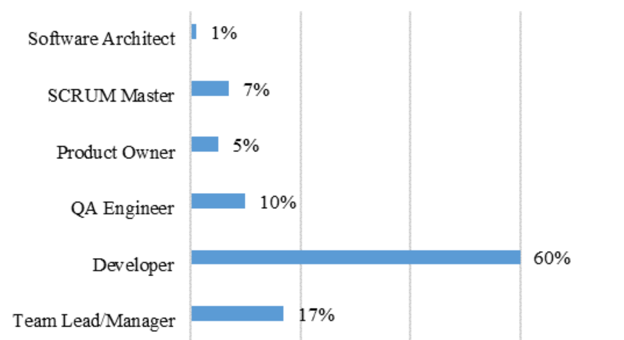


Figure 3. Team role in the project

60% indicated themselves as developer, 17 % as team leader/ manager, 10% as QA engineer, while the remaining 13% of the respondents are active in other roles, such as Scrum master, product owner and software architects. The majority of respondents (85%) are working on development project and 12% on maintenance project (Figure 4)

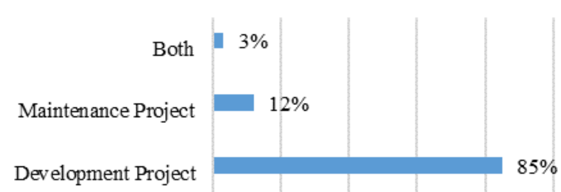


Figure 4. Main team assignment

TABLE I. AGILE TEAMWORK PRODUCTIVITY INFLUENCE FACTORS-QUESTIONNAIRE RESULTS FROM FREQUENCY ANALYSIS:ARITHMETIC MEAN, STD. DEV AND INTERNAL CONSISTENCY TEST

Sl	Factors	Description	Mean	Std. Dev	Cronbach's Alpha
1.	Staffing	The right persons should be selected	2.73	.482	.882
2.	Size of team	Small and mixed team	1.93	.362	.885
3.	Project Complexity	Database size, architecture, complexity of interface to other system, code, interface complexity to hardware and software, logical problem	1.97	.551	.883
4.	Team Leadership	Shared leadership can be shown by several team members	2.57	.621	.880
5.	Mutual performance monitoring	Being aware of other team members' performance	2.37	.637	.881
6.	Backup Behavior	Being available to assist other team members when needed	2.32	.651	.879
7.	Team orientation	Assigning high priority to team goals and participating willingly in all relevant aspects of the team	2.48	.651	.881
8.	Adaptability	Response to changing conditions, internal or external	2.45	.622	.883
9.	Feedback	Giving, seeking, and receiving of information among team members	2.48	.624	.880
10.	Mutual trust	Shared belief that team members will perform their roles and protect the interests of their team-mates	2.62	.524	.881
11.	Coordination	Team members executing their activities in a timely and integrated manner	2.75	.474	.880
12.	Communication	Exchange of information between two or more team members in the prescribed manner and using appropriate terminology	2.65	.606	.882
13.	Team members are appreciated for working long hours	Team incentive for working overtime to finish a job	1.72	.761	.884
14.	Team reward	Overtime reward for working extra time (then or later)	1.93	.733	.882
15.	Adequate technical training for team	Offering appropriate training for new technologies	2.57	.563	.880
16.	Adequate team skills training for team	Communication, organization, interpersonal, etc.	1.78	.415	.885
17.	Turnover	Staff leave or entry in the project team	1.93	.733	.884
18.	Key personnel Stayed throughout the project	Impact of personnel turnover on team	2.37	.610	.882
19.	Reuse	Software products, processes, artifacts, including components, frameworks, and software product line	2.38	.585	.879
20.	Goals	Establishment is critical for the success of the team	2.37	.637	.879
21.	Intra group wage inequality	Fair wage	1.90	.775	.883
22.	Dealing Cultural differences	Cultural differences among offshore organization	2.15	.659	.882
23.	Self-management	Most work-related decisions are made by the members of team rather than manager	2.13	.430	.887
24.	Task variety and Innovation	Team get chance to learn the different tasks the team perform to meet the workload needs of the team	2.40	.694	.877
25.	External Dependencies	Waiting for customer acceptance/for a component; interacting with external customers; publishing version of system/of data model across different environments	1.90	.511	.884
26.	Tool Usage	Use of CASE tools	2.13	.623	.880
27.	Programming Language	Programmer's experience and skills	2.13	.747	.883
28.	Schedule Pressure	The impact of intangible project pressure	1.95	.429	.884
29.	Pair Programming	Two programmers working collaboratively to develop software	1.80	.514	.883
30.	Resource constraints	e.g. timing, reliability, storage, team size, and project duration	2.37	.637	.878
31.	Team Management	Quality of management, conflict management, task assignment, administrative and formal coordination	2.55	.565	.880
32.	Motivation	To work on the project and in the company	2.57	.593	.880
33.	External project factors	Customer involvement, Customer expectation, Customer satisfaction	2.30	.696	.879
34.	Culture	Agile requires a true cultural change from plan-based approach, not only a simple change in the processes used	2.10	.796	.879
35.	Working environment	Suitability of the workplace to do creative work, collocation, e.g., windows, natural light, size of room and desk, meals provided	2.33	.629	.878

Majority of the respondents' (42%) software organization's size is small, between 30-50 people (see Figure 5). However, 30% of organizations employ 100-150, and 12% employ more than 150 people.

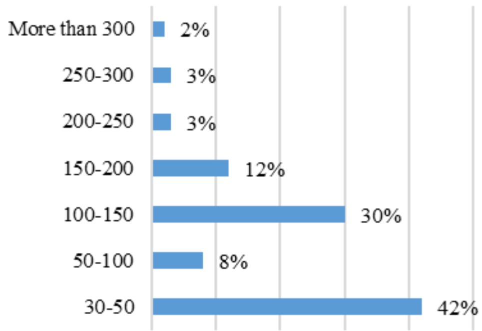


Figure 5. Size (people) of the software companies

As can be seen from Figure 6, Scrum is extensively used by the software companies. 97% indicated Scrum and 3% chose Kanban and XP. Scrum is the most popular Agile methodology also in [16][18].

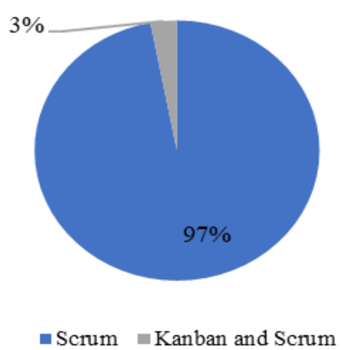


Figure 6. Most followed agile development method

According to the respondents (50%), the frequently used programming language in their organization is C# and then JavaScript followed by Java (see Figure 7).

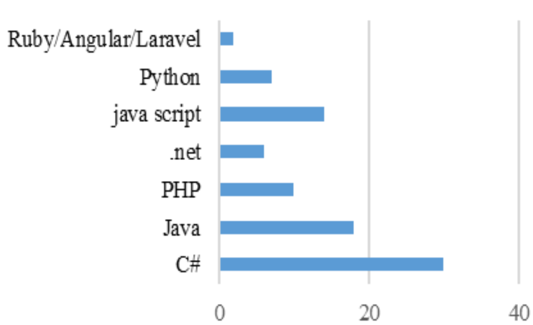


Figure 7. Programming languages use in software companies

Regarding the agile practices in use by the participating software companies, the results are well aligned with the results of similar survey [16]. Figure 8 presents the most adapted practices are daily standup meeting, release planning, stories and retrospective.

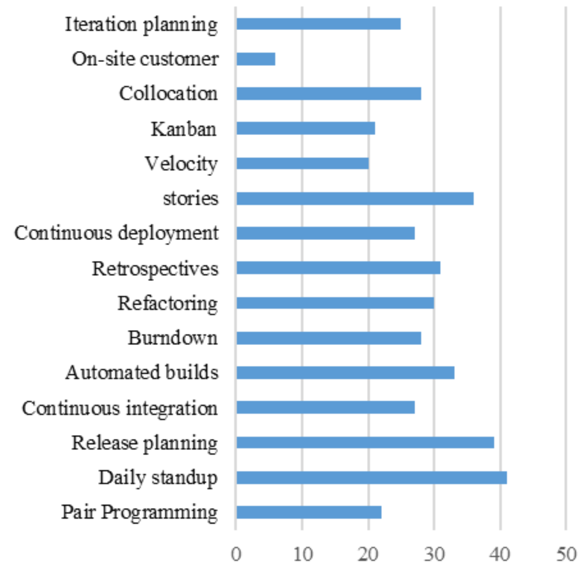


Figure 8. Agile practice adopted in software companies

Figure 9 shows that lack of management support (e.g., resource constraint, team design choice and motivation) is the main reason for failure in agile projects. In addition to this, respondents have mentioned another three more reasons in this extension of the previous study [1][40]. Integration failure, frequent change request and substantial funding crisis are mentioned by the survey respondents.

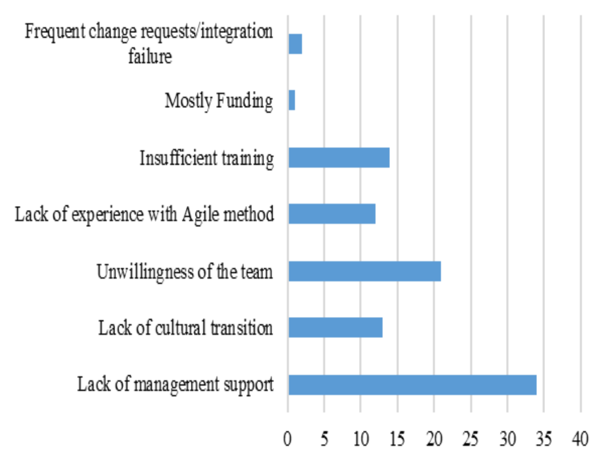


Figure 9. Main reason for failure in agile project

Lack of experience with Agile methods and the company culture are indicated as project failure reasons in similar surveys [16]. The most recent survey on Agile acceptance and success or failure project results indicate lack of experience with Agile methods and the company culture are main project failure reasons [16].

In most of the interviews, the team members could not define productivity as their performance measurement. A large number of them mentioned that team management has their own ways of measuring productivity. Although at the end of the project, the management assessed their productivity on the basis of timeliness and quality. At the same time, ten interviewees and survey respondents (Figure 10) also mentioned customer satisfaction as a criterion for measuring or perceiving productivity. Customer satisfaction is very important to software development companies in Bangladesh as a rising market for outsourced software destination. This study result also confirms latest worldwide survey studies that have shown customer/user satisfaction is the number one measure of an agile project's success [16].

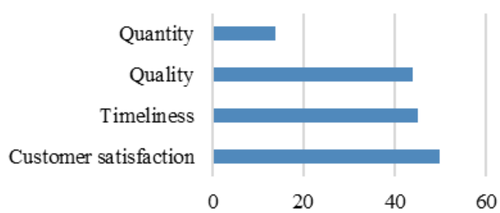


Figure 10. Criterion for measuring or perceiving productivity

According the product owner interview, dealing with cultural differences among offshore organization influences teamwork productivity. Two main reasons behind this are time and culture differences.

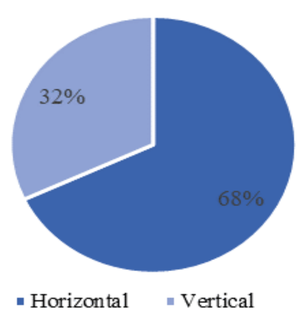


Figure 11. Organizational structure

Sometime it becomes difficult to keep contact with the offshore client on urgent issues due to time difference between places. Moreover, offshore client's expectations are different, both in terms of their general culture and their views on life and work. Project developed within western cultures are different from eastern cultures. For example, daily traffic condition consumes most of the working time in Bangladesh, which makes the developers less motivated. Since staff are not rewarded enough for working long hours. However, schedule pressure can be easily dealt with overtime working because it is inexpensive in Bangladesh.

Five interviewees (project leads and managers) mentioned that culture is a big barrier for working in an agile team. Even though it is not one of the most influential factors mentioned by survey respondents. The survey result shows (Figure 11) that the participating software companies' organizational structure and coordination are primarily horizontal (68%), where coordination processes are usually provided by an individual team member who communicates directly with other members or users on a one-to-one basis [18][41]. On the other hand, vertical coordination (32%) is usually implemented through project managers. The horizontal structure of agile involves self-organizing teams that work in an iterative fashion and deliver continuous additional value directly to customers [41]. Although the practice of self-organized teams conflicted with the cultural responses of social hierarchy. According to Balasubramaniam et al., Social hierarchy recommends a top-down approach to decision making, which is different from a participatory approach and hinders teamwork [41]. In Eastern culture, workplace hierarchies are common practice of being superior to others in authority, power, or status that are commonly accepted by subordinates. Team members look for clear instructions and accept their supremacy, also their own dependency on the superiors.

Based on results found in the interview and survey of this study, it is perceived that social hierarchy is embedded in Bangladeshi software organizations and affects the implementation of agile principle. In agile development, communication links together all other teamwork processes. Therefore, regular and informal meeting should take place among team members. The survey result shows that project/team management has more influence on productivity than the self-management. Beside that most popular agile practice among the participating companies is Scrum and there is no such role of project manager. In the agile approach, team should be self-managed and work is coordinated by the team members [23].

From this scenario, it is evident that, even though most followed organizational structure is horizontal, social hierarchy culture significantly influences agile teamwork productivity. It is because the way team members communicate with team and customers, and more often to respect official hierarchy/top management (cultural norm in Bangladesh), communication occurred between members at the same levels of the organizational hierarchy [41].

In addition, sometimes language barrier hinders communication. Cultural transitioning from individual work to self-management team requires a reorientation not only by developers but also by management. This changeover of organizational culture and institutional process take time and resources. These begin from changes of individual perception and for this reason, project managers prefer fresher as a team member. Their software companies like to groom up with several activities such as training, community, and conference than changing traditional mind set up of the team members.

Stable teams are associated with higher productivity, so avoiding changing team members to keep key personal throughout the project has great influence on productivity (Table I and Figure 12). Sustainable pace is an essential part of agile development, and only by working regular hours at a reasonable level a team can produce good flow of work [6]. Productivity grows over time through the development of the teamwork practices by team learning process and not by doing overwork or compromising the quality to increase team's productivity [18]. Moreover, teams are not rewarded enough for working long hours (Table I). This study findings also indicate that schedule pressure has less impact on productivity and; timeliness and work quality are the most mentioned criterion for measuring or perceiving productivity (Figure 12).

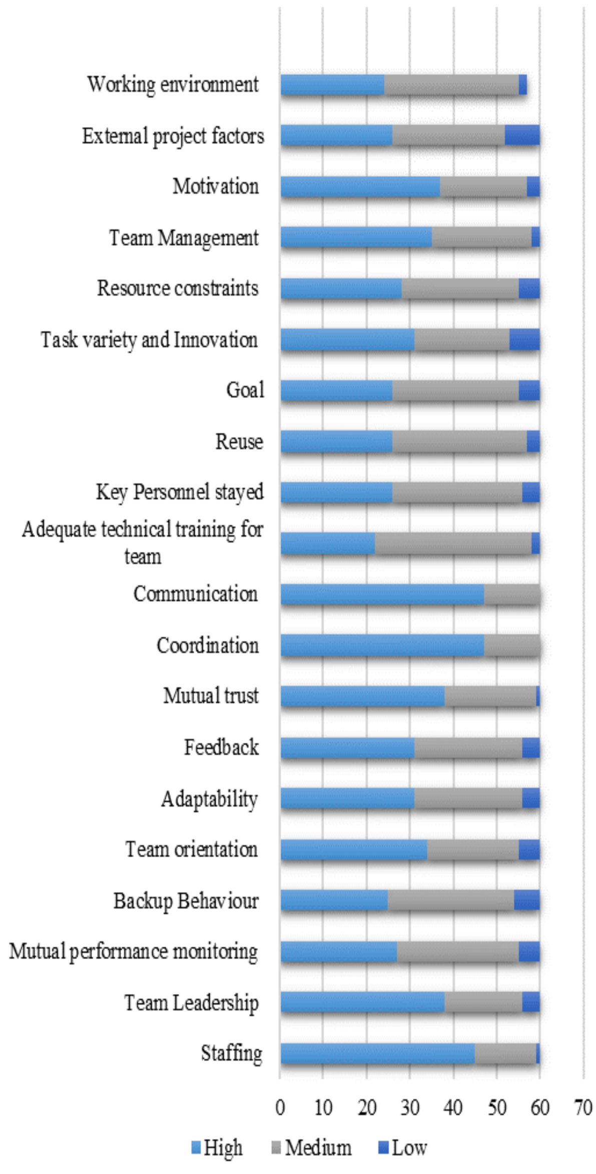


Figure 12. Agile team perception of productivity influence factors

Figure 12 provides highlights of the most influencing productivity factors that are perceived by the agile team members. This study results show that agile teamwork is highly dependent on team effectiveness. Offshore clients' satisfaction (external factors dependency) is very important for the organization. Team leadership and team orientation are very important for teamwork motivation. The factors impacting on agile teamwork productivity mentioned by the team members suggested that feedback, team orientation, communication, coordination and mutual trust improve team effectiveness. Eventually, this will enable team to learn how to effectively manage relationship within team in order to become more productive.

In sum, the study results show that some traditional productivity factors (from Table I) are still influential factors to agile software development teamwork productivity, even with the adoption of the agile practices. A transition to self-managing agile team is one of the biggest challenges when introducing agile development in Bangladeshi culture. Agile implementation needs the mindset change of all the team members; investment in training and learning oriented activities will make the Agile team members more productive.

V. QUALITATIVE MODELLING OF AGILE SOFTWARE DEVELOPMENT TEAMWORK PRODUCTIVITY

Software development productivity is a function of complex set of "hard" and "soft" factors [2]. Most of the data required to understand the development and dynamics needed to determine the factors that influence agile teamwork productivity mainly are concerned with soft factors [4][13]. The SD approach is capable of incorporating the soft factors, which can have an important influence on the agile teamwork. Soft factors such as productivity, motivation, team management efficiency, customer satisfaction, skillfulness and team effectiveness may be included and represented visually as a CLD. In the following section, the complex inter-related structure of different influence factors is modelled using qualitative SD approach (CLD). Vensim [42], free SD modelling software package is used for this research work.

A. Causal Loop Diagram (CLD)

Each factor that affects agile teamwork productivity is itself affected by other factors [18]. Some factors may be the reaction of the same action [31]. In system dynamics, this reaction is called feedback. There are two types of feedback – reinforcing feedback and balancing feedback. Sometimes a feedback (or a reaction) does not occur immediately – the process contains delays. Dynamic system can be drawn as a model with circles of causality – including actions, feedbacks and delays [2].

Technically, a CLD consists of words or phrases, which are linked by curved arrows, each of which has attached sign (positive or negative) and occasionally a time delay symbol [2]. The arrow represents a causal relationship between two factors. The sign is symbolized by '+' indicating the two related variables change in the same direction, or '-' showing the two linked variables change in two different directions; and the time delay is shown by '//' crossing the arrow.

Overall conceptual model (influence diagram) of agile teamwork productivity is presented in Figure 13, which is developed by identified factors in research Step 1 and then linking these factors to show their influences. For each of these links, the relationship is indicated as positive in the case of the same variation for both connected factors and negative for the opposite case.

The SD is based on the ground that these underlying influences are crucial to project management and need special attention [22]. This resulting model is used to understand and explain factors and feedback relationships between the influencing factors over time.

B. Analysis of causal links between agile teamwork productivity influence factors

A CLD of an identified research problem is developed by already established ideas and research in addition to the researcher's mental model [2]. This section summarizes some of the most influential productivity factors based on causal loop diagrams developed by [5][21] for software development productivity. The elaborated relationships are presented with the help of CLD for some essential factors from Table I, in order to get a basic understanding of the feedback concepts. To keep the readability of CLD, it has been divided into two sub models (Figure 14 and Figure 15).

Starting with actual teamwork productivity in Figure 14, which is positively influenced by potential teamwork productivity and a number of complex factors as identified from the study. The proposed agile teamwork productivity model is based on the following state:

Actual productivity = potential productivity – losses due to faulty processes [21]. Where losses due to faulty process mainly refer to communication overhead and motivational losses. Potential productivity represents the best possible use of resources and the maximum level of productivity the team can produce.

As seen in Figure 14, team morale positively influences work quality, as highly motivated team generate fewer errors and less rework. Expecting higher quality and high team morale, in turn increase the customer satisfaction [34]. The result of increased customer satisfaction is a decrease in external factor's influence on teamwork productivity and thus have an indirect (positive) effect on motivation. Customer satisfaction is one of the indicators of productivity [6] and less external factors along with team morale and motivation positively influence the overall teamwork productivity. Working environment, reward and salary directly influence motivation. Goals set by team management is a future condition to motivate them to work towards its accomplishment and morale development in the team [43]. High level of team morale to the project will increase the development motivation. The impact factors to the level of motivation include the relationships of the team, team management, individual salary, working environment, reward etc.

Figure 15 compiles the cause-effect relationships connecting team effectiveness, team management, motivation, learning factors with teamwork productivity.

Agile software development emphasizes teamwork in self-organizing teams more than traditional development methods do. It is useful to learn how team works effectively in order to better understand the factors, together, influence the productivity of agile teamwork. This study considers an adapted form of Salas Big Five teamwork theory [26] and the Dickinson and McIntyre model of team effectiveness [23] for the betterment of agile teamwork. These two models focus on team effectiveness, and mainly on internal components of teamwork. At the same time, both of the models consider the teamwork activity as a learning loop in which teams are identified as self-managed, adaptable and dynamically changing over time [23]. These self-managed agile teams are usually responsible for managing, monitoring and executing their own tasks. It also requires a double-loop learning, which is a characteristic of self-managing agile teams to change underlying values and assumptions [26]. The findings from the survey include the productivity influence factors that also comply well with Salas [26] and Dickinson and McIntyre's [23] teamwork components.

Figure 15 shows how the team effectiveness is built within a team and how it affects the engagement of the team in learning-oriented activities (learning factors).

Mutual trust concept is based on shared belief that the team members feel accepted and respected for their feedback. Without sufficient trust, team members will spend time and energy protecting, checking, and inspecting each other as opposed to mutual performance monitoring [44]. It is evident that trust is a prerequisite for shared leadership, feedback, and communication. Team members may not be willing to participate or share information if they fear being perceived for incompetent performance. The degree of the mutual trust, adaptability, team orientation, coordination and communication can be impacted by the experience of working together. More the team members understand each other, higher the ability of the team to identify problem in a short time frame and hence increase teams' potential productivity [5].

The team inspiration to engage with the learning factors is positively related to the team effectiveness in regard to team orientation and mutual performance monitoring and feedback present in an agile team. This perception is represented in the CLD (Figure 15) by the factor motivation, which offers support to team members to overcome the fear that arises when they face difficult situations. Therefore, the higher the level of motivation, the more secure team members feel, and the more willing they become to involve in learning-oriented activities. As the project proceeds, the team members increasingly engage in learning activities, they interact and coordinate more, hence the potential team productivity increases.

Dickinson and McIntyre model suggests that team leadership and team orientation promote team members' capability to monitor their team members' performance

[23]. Consequently, it enhances team effectiveness, which leads to improved team productivity.

The team effectiveness, including team management efficiency, are both influenced by skillfulness and might be enhanced by training. Training strengthen the teams' process knowledge, which in turn improves team members' skills and capabilities. Teams' expertise is further influenced by individual learning, which is characterized by individual work experiences [45]. Individual learning positively influences organizational learning, which can be further created through shared experiences [45].

The team motivation is also affected by the behavior of the team management. According to Melo, agile team management is the most important factor in achieving agile team productivity [5]. A supportive team management tends to provide constructive feedback and encourage team to involve on task variation. However, a team management, which follows social hierarchy, promote a top-down approach to decision making, as opposed to a participatory approach, significantly influenced the way team members communicated with each other. Under this kind of team management, agile team members will avoid any unwanted situation where they can face problem and restrain from learning-oriented activities. As a result, team productivity decrease, the team management efficiency decrease, indicating a perceived need for team/technical training, as represented in Figure 15.

Another factor that influences skillfulness is pair programming, which is one of the key practices influencing team productivity [16][18]. However, this factor is not encouraged in Bangladeshi software companies. Management does not want to engage two resources for single work due to increase in expenses. It is mostly practiced by the developers when they need assistance to complete a difficult work.

Getting the right person with suitable skills and knowledge for an agile team is a difficult job for the software companies in Bangladesh. Staffing (right person selected) happened to be as one of the most important factors impacting teamwork productivity, as Table I and Figure 12 show. Consequently, team design choice becomes a significant influencing factor for agile teamwork productivity (Figure 13). It affects the amount of knowledge that team members must apply to improve the team task (Figure 15).

VI. CAUSAL LOOP DIAGRAM VALIDATION

"All models are simplified representations of reality. Therefore, they are wrong by definition, yet they may be useful for particular purposes." [2]. This qualitative model's purpose is to provide a better understanding of the factors and mediators that influence agile teamwork productivity. At this stage, its validity is assessed based on how clearly it can explain insight dynamics of the system. Exact precision of SD forecasting models is not expected. For this reason and to recognize the rightness of the diagrams, the following criteria have been used to study the fitness of the causal diagrams to fulfil the objectives of the study (adapted from Coyle [46], p46):

- Have the purpose and the target audience for the diagram carefully chosen?
  The target shown in all diagrams is the influence of the productivity factors. The diagrams show the linkage between the factors up to the stage where it influences the team productivity.
- Are the factors, which it includes consistent with the purpose?
  All the factors included in the CLD affect the productivity of the agile teamwork, which is the purpose of the study (Table I).
- The objective of system dynamics is policy analysis against a range of circumstances, so are the policies clearly shown in the diagram?
  The objective of the causal diagrams shown in this study is to show how the factors influence the productivity as a first step toward developing a complete quantitative model, which will quantify these relationships in future work.

To clarify the causal relationship between the factors and teamwork productivity and as a step towards building the qualitative model, most influential factors have been concluded with two sub-causal analysis diagrams. That explains the relationship between the factors and the mediator(s), which have direct influence on the agile teamwork productivity using prior theoretical knowledge extracted from the literature, interviews and survey.

VII. LIMITATIONS OF THE STUDY

There are a number of limitations to this study. First, this study was limited to 60 respondents and 12 interviewees from 18 software companies. It was challenging to get access to more software companies due to time constraint and its access to appropriate resources was limited.

Respondents were carefully chosen from different roles within the agile team in order to get different perspectives of productivity in the context of Bangladesh software Industry.

Another limitation of this study is the agile team members' perceptions used as a response. However, with survey, this study relies on what the respondents provided to the researcher. It is possible that the respondents' perception may change and be different after the end of the project. To minimise the impact of this effect, the survey and interviewees' responses were compared for factors selection to include in the model. The questionnaire used for this study had been used successfully in other research and was developed after a detailed literature review [15][18]. Some of the questions were included in the survey after getting knowledge about the working conditions of software companies in Bangladesh from the interview sessions.

This CLD models certainly contains inherent limitations and is not complete because it only focuses on a limited number of important soft factors and it does not present explicitly the reinforcing/ balancing loop and delay.

The scope of this empirical findings considers the Bangladeshi software companies as a case study, which can

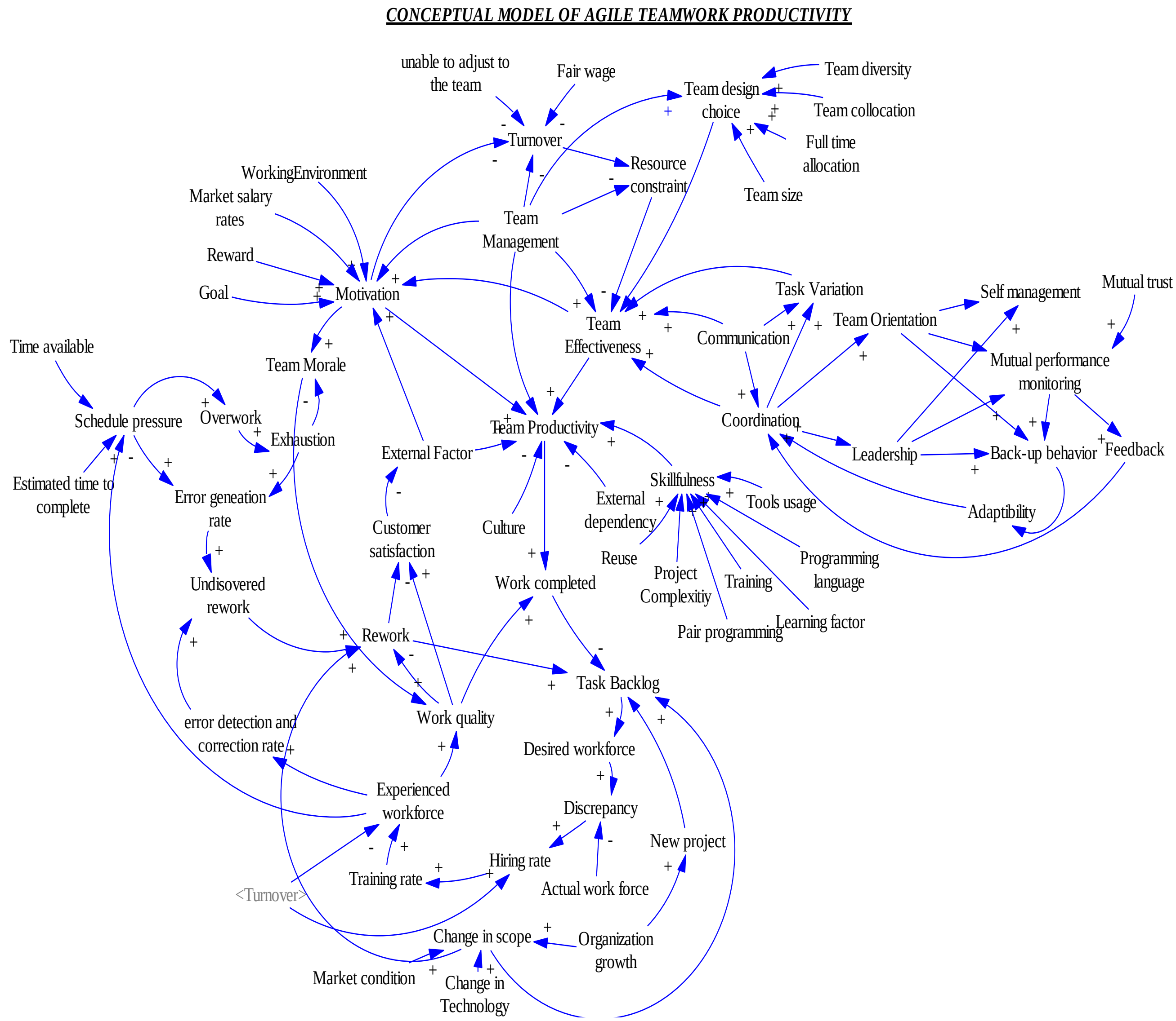


Figure 13. Overall conceptual model of Agile teamwork productivity

in turn make the research results beneficial to these companies. All the data used in this study is collected from the software companies who have voluntarily participated in this research. Therefore, findings from this study should be generalized with caution. While the findings may be specific to the contexts studied, analytic generalization could facilitate the application to other types of culture, background and environment.

VIII. CONCLUSION AND FUTURE WORK

The productivity of the development team is important for successful software project. The agile team, which is the most dynamic element and the human input in the software development industry, gain more interest to study their productivity. This research aimed to present a system dynamics based approach to model agile teamwork productivity. In order to achieve this, the main factors that affect teamwork productivity were determined via two-phased approach, where in the first phase a systematic literature review, interview and survey of different agile teams were conducted to collect and select impacting factors, and they were evaluated and ranked to identify the most influential ones. The second phase involved the development of qualitative SD model (causal loop diagram) of agile teamwork highlighting the different influencing factors. The findings of this stage are the main influencing factors, which are motivation (external factors, customer satisfaction), team effectiveness (communication, coordination, mutual trust, leadership) and team management (staffing, Key personnel Stayed throughout the project, team design choice). Moreover, this study used two team effectiveness models- Salas [23] and the Dickenson McIntyre [12] model for better understanding and analyzing inter aspects of agile team. The most cited and influential factors were: Coordination, Communication, mutual trust and staffing (right person selected for the team). These factors were the most important for effective teamwork and team management in the agile teamwork productivity.

According to study results, lack of management support is found to be the most mentioned reason for any failed agile project. The most followed organizational structure is horizontal and most followed agile method is Scrum. In addition, this study finds that due to social hierarchy culture influences, self-manage agile team may not fully fit in their organization. This factor also hinders agile transformation from plan driven to self-managed agile team.

As a future work, survey (statistical) data will be used to estimate influencing factors (hard and soft) to quantify the effects of the factors on productivity. The outputs will be applied in quantitative modelling of team productivity via SD approach and assess the ability of the model to duplicate historical data when measurable data are available.

The proposed CLD model will be used as a basis for developing stock and flow model of SD method. Further research need to be conducted for qualitative in-depth studies of the causes behind the results in regard to certain factors and the model against a real-world agile software development project. The proposed SD model of agile teamwork productivity will provide more strategic observance and competence about the effectiveness of different managerial policies based on non-straight forward cause-effect relationships hidden in the system.

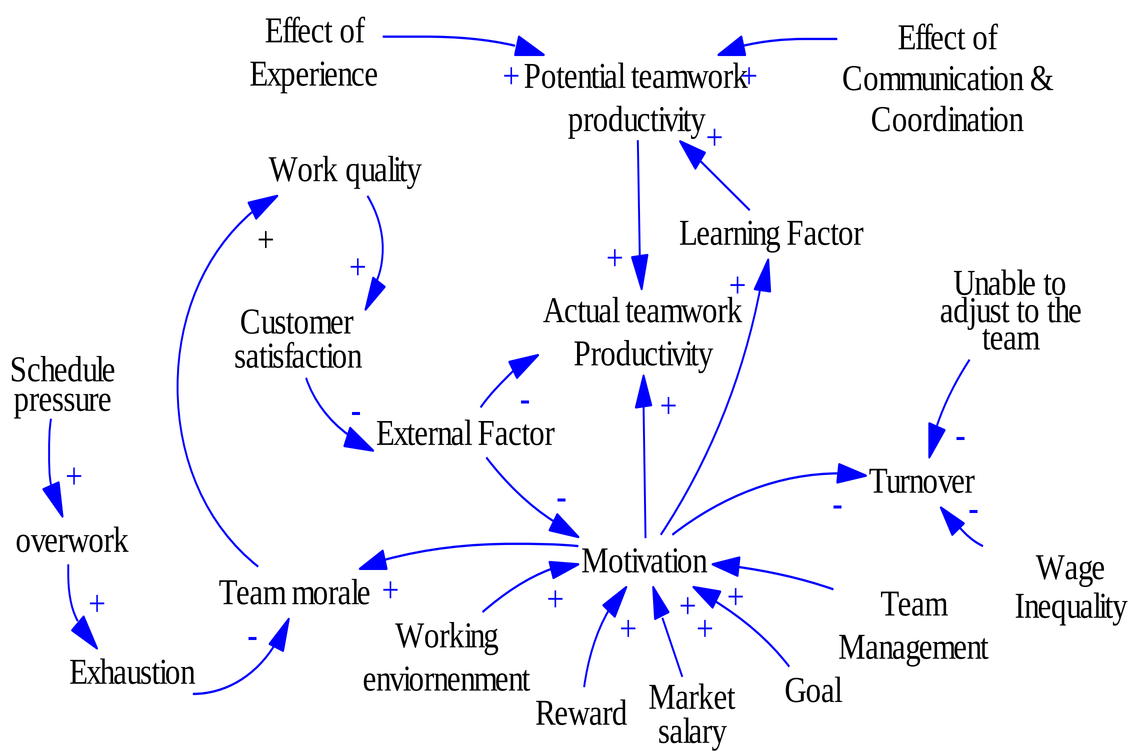


Figure 14. Causal loop diagram ‘the influence of motivation on productivity’ sub-model

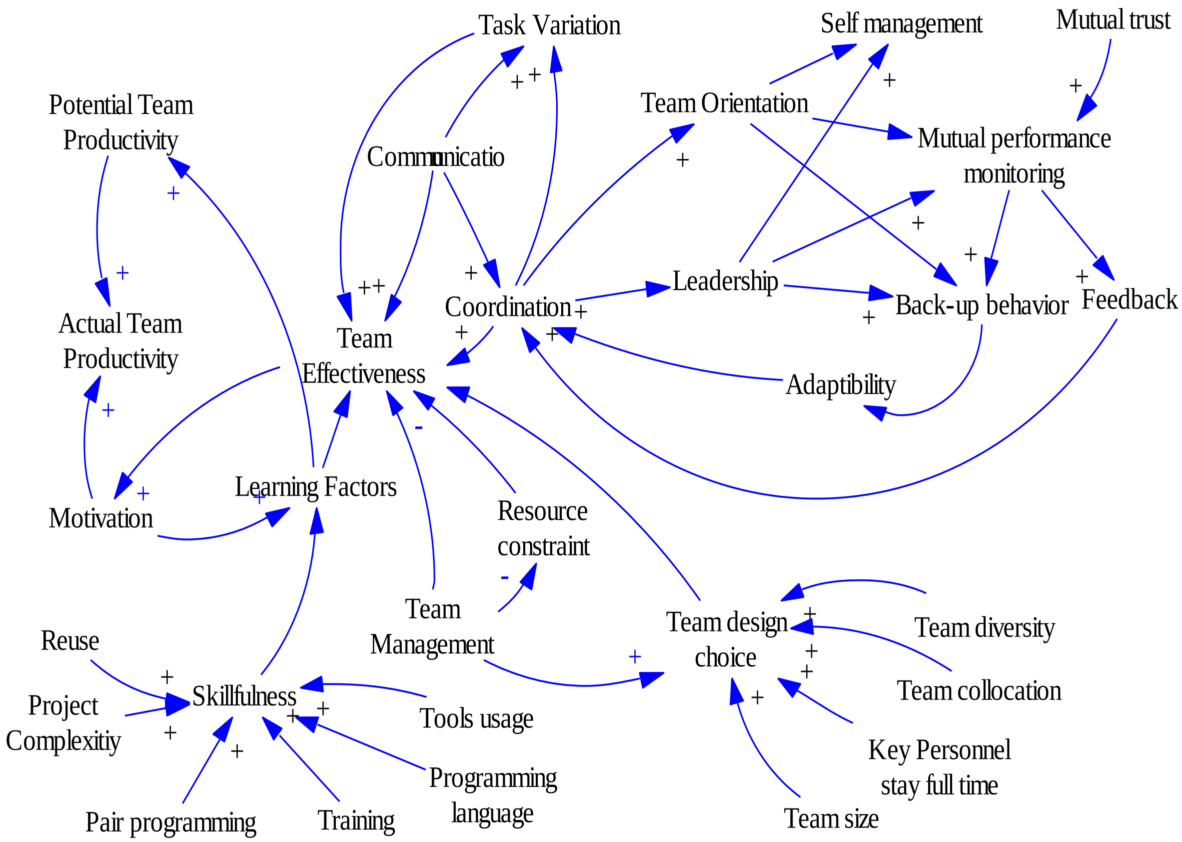


Figure 15. Causal loop diagram 'the influence of team effectiveness and team management, on productivity' sub-model

ACKNOWLEDGMENT

We would like to thank the companies and their employees who contributed to this research. This research is supported by ICT Division, Ministry of Posts, Telecommunications and Information Technology, Bangladesh. 56.00.0000.028.33.079.17-223, 20.06.2017.

REFERENCES

[1] I. Fatema and K. Sakib, "Analyse Agile Software Development Teamwork Productivity using Qualitative System Dynamics Approach," The Twelfth International Conference on Software Engineering Advances (ICSEA 2017), Oct. 2017, pp. 60-66, ISBN: 978-1-61208-590-6

[2] J. D. Sterman, "Business Dynamics: System Thinking and Modeling for a Complex World," Boston: The McGraw- Hill, 2000.

[3] A. Trendowicz and J. Münch, "Factors Influencing Software Development Productivity—State-of-the-Art and Industrial Experiences," Advances in computers, vol. 77, pp. 185-241, Dec. 2009.

[4] G. P. Sudhakar, A. Farooq, and S. Patnaik, "Soft factors affecting the performance of software development teams," Team Performance Management: An International Journal, vol. 17, no.3, pp. 187-205, 2011.

[5] X. Kong, L. Liu, and D. Lowe, "Modeling an agile web maintenance process using system dynamics," In 11th ANZSYS/Managing the Complex V conference, ISCE Publishing, Christchurch, NZ. Dec. 2005.

[6] C. D. O. Melo, D. S. Cruzes, F. Kon, and R. Conradi, "Interpretative case studies on agile team productivity and management," Information and Software Technology, vol. 55, pp.412-427, Feb.2013.

[7] K. Petersen, "Measuring and predicting software productivity: A systematic map and review," Information and Software Technology, vol. 53, pp.317-343, Apr.2011.

[8] Y. Ramírez and D. Nembhard, "Measuring knowledge worker productivity: A taxonomy," Journal of Intellectual Capital, vol. 5, no. 4, Dec. 2004, pp. 602–628.

[9] How to estimate the required size of agile teams? ISBSG. [Online]. Available from http://isbsg.org/wp-content/uploads/2017/08/Estimate-Agile-team-size-v1.0.pdf [retrieved: May, 2018].

[10] Productivity measurement in Agile teams – why is it so difficult? [Online]. Available from http://isbsg.org/isbsg-agile/ [retrieved: May, 2018].

[11] T. DeMarco and T. Lister. Peopleware. Productive Projects and Teams. Dorset House Publishing, 1987.

[12] T. Dingsøyr and T. Dybå, "Team effectiveness in software development: Human and cooperative aspects in team effectiveness models and priorities for future studies," Proceedings of the 5th International Workshop on Co-operative and Human Aspects of Software Engineering. IEEE Press, 2012, pp. 27-29.

[13] V. Lalsing, S. Kishnah and S. Pudaruth, "People factors in agile software development and project management," International Journal of Software Engineering and Applications, vol. 3, no. 1, pp. 117, 2012.

[14] S. C. Misra, V. Kumar, and U. Kumar," Identifying some important success factors in adopting agile software development practices,"Journal of Systems and Software, vol. 82, no. 11, pp. 1869-1890, 2009.

[15] J. M. Verner, M. A. Babar, N. Cerpa, T. Hall, and S. Beecham, "Factors that motivate software engineering teams: A four country empirical study,". Journal of Systems and Software, vol. 92, June. 2014, pp. 115-127.

[16] VersionOne: 12th annual state of Agile survey (2017). [Online]. Available from: https://explore.versionone.com/state-of-agile/versionone-12th-annual-state-of-agile-report. [retrieved: May, 2018]

[17] T. Dybå and T. Dingsøyr, "Empirical studies of agile software development: A systematic review", Information and software technology, vol. 50, no. (9-10), pp. 833-859, 2008.

[18] C.O. Melo, "Productivity of agile teams: an empirical evaluation of factors and monitoring processes," Ph.D. dissertation, Universidade de São Paulo, 2015.

[19] T.K. Abdel-Hamid and S. Madnick, "Software productivity: potential, actual, and perceived,". System Dynamics Review, 5(2), pp. 93-113, June. 1989.

[20] J. M. Lyneis and D. N. Ford, "System dynamics applied to project management: a survey, assessment, and directions for future research," System Dynamics Review, vol. 23, no. 2-3, pp. 157-189, Jun. 2007.

[21] Abdel-Hamid, T. (1991) "Software Project Dynamics, an Integrated Approach", Englewood Cliffs, NJ. Prentice-Hall, Inc., 1991.

[22] A. Rodrigues and J. Bowers, "The role of system dynamics in project management," International Journal of Project Management, vol. 14, no. 4, Aug. 1996, pp. 213-220.

[23] N. B. Moe, T. Dingsøyr, and T. Dybå, "A teamwork model for understanding an agile team: A case study of a Scrum project," Information and Software Technology, vol. 52, pp. 480-491, May. 2010.

[24] L. Cao, B. Ramesh, and T. Abdel-Hamid, "Modeling dynamics in agile software development," ACM Transactions on Management Information Systems (TMIS), (1:1), p. 5, 2010.

[25] K.E. Van Oorschot, K. Sengupta, and L. van Wassenhove,"Dynamics of agile software development" In Proceedings of the 27th International Conference of the System Dynamics Society, Albuquerque, New, Mexico, USA. Dec. 2009.

[26] T. Dingsøyr and Y. Lindsjørn, "Team performance in agile development teams: findings from 18 focus groups," International Conference on Agile Software Development, Springer Berlin Heidelberg, June. 2013, pp. 46-60.

[27] A. Teh, E. Baniassad, D. V. Rooy, and C. Boughton, "Social Psychology and Software Teams: Establishing Task-Effective Group Norms," IEEE Software, vol. 29, no.4, pp.53–58, Jul. 2012.

[28] B. Tessem and F. Maurer, "Job Satisfaction and Motivation in a Large Agile Team," In International Conference on Extreme Programming and Agile Processes in Software Engineering, Springer Heidelberg, vol. 4536, pp. 54–61., 2007.

[29] N. B. Moe, A. Aurum, and T. Dybå, "Challenges of shared decision making: A multiple case study of agile software development," Information and Software Technology, vol. 54, pp.853–865, Aug. 2012.

[30] B. Barry, Software Engineering Economics, New York, vol. 197, NY: Prentice-Hall, 1981.

[31] F. Nasirzadeh and P. Nojedehi, "Dynamic modelling of labour productivity in construction projects," International Journal of Project Management, vol. 31, no. 6, Aug. 2013, pp. 903-911.

[32] W. Scacchi, "Understanding and improving Software Productivity," Advances in Software engineering and Knowledge engineering, 2005.

[33] A. G. Rodrigues and T. M. Williams, "System dynamics in project management: assessing the impacts of client behaviour on project performance," Journal of the Operational Research Society, vol.49, no. 1, pp. 2-15, 1998.

[34] L. L. R. Rodrigues, N. Dharmaraj, and B. R. Shrinivasa Rao, "System dynamics approach for change management in new product development," Management Research News, vol. 29, no. 6, Aug. 2006, pp. 512-523.

[35] M. J. Mawdesley and S. Al-Jibouri, "Modelling construction project productivity using systems dynamics approach," International Journal of Productivity and Performance Management, vol. 59, no.1, Dec. 2009, pp. 18-36.

[36] J.S. Lurey and M.S. Raisinghani, " An empirical study of best practices in virtual teams," Information and Management, vol. 38, no. 8, pp. 523- 544, 2001.

[37] I. Fatema, "Agile teamwork productivity influence factors," Jan. 2017. [Online] Available https://goo.gl/forms/I5xGdQGqFMk9he5f2. [retrieved: May, 2018].

[38] A. S. Ami, A. Imran, A. U. Gias, and K. Sakib, "Effects of Internship on Fresh Graduates: A case study on IIT, DU students" doi: 10.13140/RG.2.2.13745.68967. [Online]. Available from https://www.researchgate.net/publication/323393923_Effects_of_Internship_on_Fresh_Graduates_A_case_study_on_IIT_DU_students. [retrieved: May, 2018].

[39] V. Sridhar, D. Nath, R. Paul, and K. Kapur, "Analyzing factors that affect performance of global virtual teams," In Second International Conference on Management of Globally Distributed Work, pp.159-169, July. 2009.

[40] I. Fatema and K. Sakib, "Factors Influencing Productivity of Agile Software Development Teamwork: A Qualitative System Dynamics Approach," In 24th Asia-Pacific Software Engineering Conference (APSEC), IEEE, 2017, pp. 737-742, doi: 10.1109/APSEC.2017.95.

[41] B Ramesh, L Cao, J Kim, and K Mohan, "Conflicts and complements between eastern cultures and agile methods: an empirical Investigation," European Journal of Information Systems, vol. 2, pp. 26, May. 2017, doi:10.1057/s41303-016-00230.

[42] Ventana Systems Inc, [Online]. Avilable from http://vensim.com/ [retrieved: May, 2018].

[43] H. Tohidi,"Teamwork productivity and effectiveness in an organization base on rewards, leadership, training, goals, wage, size, motivation, measurement and information technology," Procedia Computer Science, Vol. 3, pp. 1137-1146. 2011.

[44] N. B. Moe and T. Dingsøyr, "Scrum and team effectiveness: Theory and practice," International Conference on Agile Processes and Extreme Programming in Software Engineering, Springer Berlin Heidelberg, Jun. 2008, pp. 11-20.

[45] E. Lizeo, “A dynamic model of group learning and effectiveness,” In System Dynamics Society Conference. July. 2005.

[46] R. G. Coyle, “System dynamics modelling: a practical approach,” Chapman and Hall: London. vol. 1, 1996

Evaluating the Accuracy of Estimates: the Case of Model-based COSMIC Functional Size Estimation

Luigi Lavazza

Dipartimento di Scienze Teoriche e Applicate
Università degli Studi dell'Insubria
Varese (Italy)
Email: luigi.lavazza@uninsubria.it

***Abstract*—Functional Size Measurement is widely used, especially to quantify the size of applications in the early stages of development, when effort estimates are needed. However, the measurement process is often too long or too expensive, or it requires more knowledge than available when development effort estimates are due. To overcome these problems, early size estimation methods have been proposed, to get approximate estimates of functional measures. In general, early estimation methods adopt measurement processes that are simplified with respect to the standard process, in that one or more phases are skipped. So, the idea is that you get –at a fraction of the cost and time required for standard measurement– size estimates affected by some estimation error, instead of accurate measures performed following the longer and more expensive standard measurement process. In this paper, we consider some methods that have been proposed for estimating the COSMIC (Common Software Measurement International Consortium) size of software during the modeling stage. We apply the most recent methodologies for estimation accuracy, to evaluate whether early model-based estimation is accurate enough for practical usage. The contribution of the paper is twofold: on the one hand we provide a reliable evaluation of the accuracy that can be obtained when estimating the functional size of software applications based on UML models; on the other hand, we get indications concerning the effectiveness and expressiveness of recently proposed accuracy estimation methods.**

Keywords–Functional size measurement; COSMIC Function Points; Measurement process; Functional size estimation; Accuracy estimation.

I. Introduction

Functional Size Measurement (FSM) is widely used. Among the reasons for the success of FSM is that it can provide measures of size in the early stages of software development, when they are most needed for cost estimation. However, FSM requires that the functional requirements of the application to be measured are available in a complete and quite detailed form. Often, this is not possible in the very early stages of development. Therefore, to get measures also when requirements are still incomplete or still defined at a coarse level of detail, size estimation models have been proposed.

When applying a size estimation method, the method –being applied to incomplete or not thoroughly detailed software specifications– requires less time and effort than the standard measurement process. However, the size estimates so obtained contain some estimation error. In general, we are ready to accept a relatively small estimation error in exchange of being able to get size estimates without having to apply the standard measurement process. On the contrary, an excessively large estimation error would defeat the very reason for measuring. Hence, we are interested in knowing the likely accuracy of measure estimates. To this end, we need reliable methods to evaluate the accuracy of estimates [1].

Unfortunately, it has been shown that the most popular estimate accuracy statistic, the Mean Magnitude of Relative Errors (MMRE) is flawed, in that it is a biased estimator of central tendency of the residuals of a prediction system because it is an asymmetric measure [2][3][4]. So, MMRE and similar indicators are not suitable for providing practitioners who are potentially interested in applying estimate methods with reliable information upon which they can base informed decisions.

Luckily, sound estimate evaluation methods have been proposed recently (as described in Section III). It is thus possible to apply such new methods to size estimation methods.

There are different types of FSM and many estimation methods. Here, we concentrate on the COSMIC FSM [5] –one of the most widely used methods– and on model-based COSMIC size estimation [6]. The main purpose of this paper is the evaluation of the actual accuracy of model-based COSMIC size estimation method: to this end, we use the new sound evaluation methods (described in Section III), together with more traditional statistical tools. There is no specific reason why the COSMIC FSM –among the many functional size measurement methods– is addressed here. The proposed method can be applied to evaluate the accuracy of estimating functional size expressed in other measurement methods, e.g., IFPUG Function Point Analysis.

It should be noted that the paper does not aim at introducing new COSMIC size estimation methods; rather, the goal of the paper is (re)evaluating the accuracy of the formerly [6] proposed ones. However, by applying these new evaluation methods, as a side effect we also get some indications on their expressiveness.

The paper is structured as follows. Section II briefly illustrates the COSMIC FSM, and model-based simplified COSMIC measurement methods. Section III illustrates the methods used for evaluating the accuracy of estimates. Section IV describes the application of the accuracy evaluation methods to model-based simplified COSMIC measurement, while Section V illustrates and discusses the results of the analysis. Section VI deals with threats to the validity of the study. Section VII accounts for related work. Finally, Section VIII draws conclusions and briefly sketches future work.

II. COSMIC FUNCTIONAL SIZE MEASUREMENT AND MODEL-BASED COSMIC ESTIMATION

COSMIC measurement is based on the analysis of the specification of functional user requirements (FUR). The FUR can be described in various ways, including the Unified Modeling Language (UML): functional size measurement of UML models was widely studied [7][8][9], also when FUR concern real-time applications [10]. During the initial stage of development, UML models are built, progressively incorporating more knowledge concerning the software to be developed: this results in progressively more complete and detailed specifications. More specifically, the UML modeling process can be seen as organized in the phases described in Figure 1. The more complete and detailed the UML model, the more elements needed for COSMIC measurement become available. Figure 1 shows the relationship between the UML diagrams that are made available by each modeling phase and the COSMIC measurement elements. During the initial UML modeling phases –i.e., before the complete and detailed FUR specifications are available– it is often the case that size measures are needed anyway. In such cases –not being possible to measure the COSMIC size of the application– we can think of *estimating* the COSMIC size, based on the information that is present in the available UML diagrams.

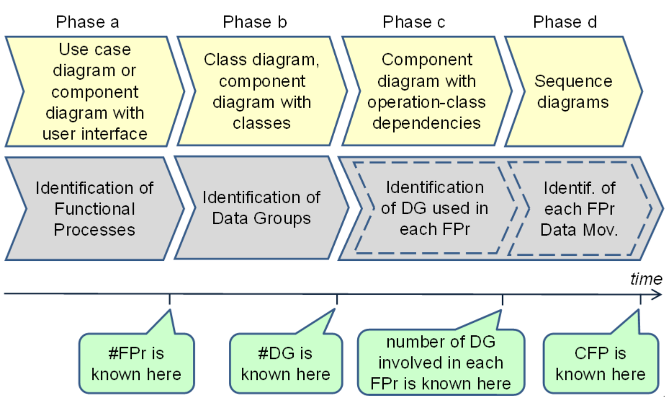


Figure 1. UML modeling process and COSMIC measurement process phases.

Specifically, del Bianco et al. proposed a few families of statistical models that can be used to estimate COSMIC size based on information derived from UML diagrams [6]. These models are described in Table I.

A first family of COSMIC size estimation models requires only the knowledge of the number of functional processes (FPrs). These models have form ECFP = f(#FPr) where ECFP is the estimated size in CFP (COSMIC Function Points), and #FPr is the number of functional processes. As shown in Figure 1, the statistical model can be built after the completion of phase a), when class or component diagrams properly specifying the user interfaces are delivered.

Another family of COSMIC size estimation models requires also that the number of Data Groups (#DG) is known. These models can be built after phase b), when UML diagrams fully describing the involved classes are delivered. The models found by del Bianco et al. involve the parameter AvDGperFPr, namely the average number of data groups per functional process, which requires that both the functional process and the data groups (i.e., classes in UML diagrams) are known.

Figure 1 shows that potentially one could use also the knowledge of the number of data groups involved in each functional process, which is available after phase c). However, no statistically significant models of this type were found.

Finally, we observe that after phase d), i.e., when the complete UML models of FUR are available, the standard COSMIC measurement process is applicable, and proper COSMIC measures –instead of estimates– can be achieved.

TABLE I. COSMIC SIZE ESTIMATION MODELS.

Name	*Formula*
avg1	$ECFP = 7.3\ \#FPr$
reg1	$ECFP = -16.5 + 6.698\ \#FPr$
avg2	$ECFP = AvDGperFPr\ 1.8\ \#FPr$
reg2	$ECFP = -64.6 + 7.63\ \#FPr + 9.71\ AvDGperFPr$
log2	$ECFP = 1.588\ \#FPr^{1.00357}\ AvDGperFPr^{1.0312}$

It is expected that models based on more information are more accurate than models based on less information.

In [6], the accuracy of the models given in Table I was evaluated based on the traditional indicators MMRE and Pred(25) (the fraction of applications for which the absolute relative estimation error is less than 0.25). The evaluation of accuracy performed in [6] indicated that models using both #FPr and AvDGperFPr (that is, models avg2, reg2 and log2) are more accurate than models based only on #FPr (that is, models avg1 and reg1). However, it has been shown that indicators based on the magnitude of relative errors are biased [11]. Hence, we repeat here (in Section IV) the analysis of accuracy using more reliable methods (described in Section III).

III. A METHOD FOR EVALUATING THE ACCURACY OF ESTIMATES

Let us first define the problem of evaluating estimates.

We measured the value of interest from n software applications; in our case the value of interest is the size of applications, measured in CFP. Accordingly, we have a set $Y = \{y_i\}$ (with $i \in [1, n]$) of observations, where y_i is the actual size of the i^{th} application, expressed in CFP.

A new estimation method P is proposed: for the n known applications, method P yields n estimates $\hat{y}_i$ with $i \in [1, n]$, and we need to evaluate the accuracy of these estimates, that is, how close to the actual y_i is the corresponding estimate $\hat{y}_i$.

In general, there are several estimation methods that can be used to carry out an estimation. Hence, the problem is not just to evaluate the accuracy of a method or model, but to compare a given method or model against other methods and models. In industrial environments, the goal is usually to compare a new method against the method currently used, to evaluate whether it could be convenient to abandon the latter in favor of the former.

A. The Mean Magnitude of Relative Errors

The most popular way of evaluating estimation accuracy has been the MMRE, the mean of the magnitude of absolute errors, which is defined as

$$MMRE = \frac{1}{n} \sum_{i=1..n} \frac{|y_i - \hat{y}_i|}{y_i} \quad (1)$$

where $y_i - \hat{y}_i$ is the estimation error (also named the residual).

MMRE has been shown to be a biased estimator of central tendency of the residuals of a prediction system, because it is an asymmetric measure [2], [3], [4]. In practice, MMRE is biased towards prediction systems that under-estimate [11].

B. The Mean Absolute Residual and the Standardized Accuracy

Shepperd and MacDonell [11] proposed that the accuracy of a given estimation method P be measured via the Mean Absolute Residual (MAR):

$$MAR = \frac{1}{n} \sum_{i=1..n} |y_i - \hat{y}_i| \quad (2)$$

Unlike MMRE, MAR is not biased [11], therefore it is preferable to MMRE.

When we need to compare a new model P with a model P_0, we have MAR_P (the MAR of P) and MAR_{P0} (the MAR of P_0). Based on these MAR values, Shepperd and MacDonell propose to compute a Standardized Accuracy measure (SA) for estimation method P:

$$SA = 1 - \frac{MAR_P}{MAR_{P0}} \quad (3)$$

Values of SA close to 1 indicate that P outperforms P_0, values close to zero indicate that P's accuracy is similar to P_0's accuracy, negative values indicate that P is worse than P_0, hence it should be rejected.

C. Baselines

When a new estimation model P is proposed, we should first establish if it is a "good enough" model, independent of possible alternative models. To this end, we compare the proposed model with a "baseline" model P_0. If there is an estimation model that is in use and is generally believed to be "good enough," the problem is establishing if P is more accurate than such model, which will act as the baseline or reference model P_0. If no reference model is available, the problem is to establish if P satisfies some minimum accuracy conditions. To this end, we use as a baseline some fairly trivial model, that requires little or no knowledge of the phenomena being estimated.

In any case, if the estimates obtained using P are less accurate than the estimates provided by the baseline P_0, we can conclude that P does not yield any improvement, at least as far as accuracy is concerned, and can be rejected, if accuracy is the only acceptance criterion. Of course we may find that P is slightly less accurate than P_0, but is much faster and cheaper than P_0, thus it could be preferable when estimates are needed as soon as possible or the estimation budget is tight.

The random model
When no obvious baseline model exists, Shepperd and MacDonell suggest to use as a referenced model random estimation, based solely on the known (actual) values of previously measured applications. A random estimation $\hat{y}_i$ is obtained by picking at random y_j, with $j \neq i$. Of course, in this way there are $n-1$ possible estimates for y_i; therefore, to compute the MAR of the random model rnd we need to average all these possible values. Shepperd and MacDonell suggest to make a large number of random estimates (typically 1000), and then take the mean $\overline{MAR_{rnd}}$. Langdon et al. showed that this is not necessary, since the average of the random estimates can be computed exactly [12].

So, a first evaluation consists in computing

$$SA = 1 - \frac{MAR_P}{\overline{MAR_{rnd}}}. \quad (4)$$

Achieving a value substantially greater than zero is clearly a sort of necessary condition that the estimation method P must satisfy, otherwise we could simply guess (instead of estimating using P) and get similarly accurate estimates, or even better ones.

Shepperd and MacDonell observed also that the value of the 5% quantile of the random estimate MARs can be interpreted like α for conventional statistical inference, that is, any accuracy value that is better than this threshold has a less than one in twenty chance of being a random occurrence. Accordingly, MAR_P should be compared with the 5% quantile of the random estimate MARs, rather than with $\overline{MAR_{rnd}}$, to be reasonably sure that P is actually more accurate than rnd.

The constant model
Lavazza and Morasca [13] observed that the comparison with random estimation is not very effective in supporting the evidence that P is a good estimation model. Instead, they proposed to use a "constant model" (*CM*), where the estimate of the size of the i^{th} application is given by the average of the sizes of the other applications, that is

$$\hat{y}_i = \frac{1}{n-1} \sum_{j \in Y - \{y_i\}} y_j \quad (5)$$

So, we can compute the MAR_{CM} of these estimates, and then compute SA, but this time comparing P with CM:

$$SA = 1 - \frac{MAR_P}{MAR_{CM}}. \quad (6)$$

Again, we require that SA is substantially greater than zero, to deem P acceptable.

Lavazza and Morasca [13] found that in real cases MAR_{CM} is quite close to the 5% quantile of random MARs. However, computing MAR_{CM} is much easier and faster than computing the 5% quantile of random MARs, thus *CM* can generally be preferred to rnd.

D. Statistical Significance

To establish if the estimates yielded by a method are significantly better than the estimations provided by another method, we need to test the statistical significance of the absolute errors achieved with different estimation methods [2]. To check for statistical significance we used the Wilcoxon Signed Rank Test [14].

The Wilcoxon Signed Rank test can be safely applied also to not normally distributed data, since it makes no assumptions about data distributions.

Via the Wilcoxon test we tested the following Null Hypothesis: "The absolute errors yielded by a model P_i are significantly less than those provided by a model P_j".

E. Size Effect

Suppose that we have two estimation methods P_1 and P_2, and $MAR_{P2} < MAR_{P1}$ (hence, $SA = 1 - \frac{MAR_{P2}}{MAR_{P1}} > 0$). We can conclude that P_2 is more accurate than P_1. Anyway, suppose that we are using P_1 and we are considering the possibility of switching to using P_2, which involves some effort, because P_2 requires some activity or data or programs that P_1 does not require. We would like to know if the improvement that P_2 offers in terms of accuracy is possibly so inconsequential as to not be worth the effort.

To judge the effect size, Shepperd and MacDonell suggest using Glass's Δ [15] or Hedges's g, which might be preferred when the sample size is small [16]. The effect size –which is scale-free– can be interpreted in terms of the categories proposed by Cohen [17] of small (≈ 0.2), medium (≈ 0.5) and large (≈ 0.8).

F. Estimate Comparison Based on Individual Absolute Residuals

Given a dataset and two models P_1 and P_2, following Shepperd and MacDonell we state that P_1 is more accurate than P_2 if $SA_{P1} > SA_{P2}$, i.e., if $MAR_{P1} < MAR_{P2}$. However, this is not the only criterion that can be used to compare the performances of P_1 and P_2. Let $\langle \hat{y}_{1,P1}, .., \hat{y}_{n,P1} \rangle$ and $\langle \hat{y}_{1,P2}, .., \hat{y}_{n,P2} \rangle$ be the estimates provided by the two models. We may say that P_1 is more accurate than P_2 if and only if there are at least $\lceil \frac{n+1}{2} \rceil$ distinct values of i such that $|y_i - \hat{y}_{i,P1}| < |y_i - \hat{y}_{i,P2}|$, that is, if P_1 provides smaller absolute residuals than P_2 in the majority of cases.

The probability that P_1 provides better results than P_2 in the majority of estimates is measured by the IARA (Individual Absolute Residual Assessment) indicator [13]. The IARA index is defined as the number of estimates for which the residuals of P_1 are less than those of P_2 divided by the total number of estimates. The statistical significance of IARA can be tested via the binomial test.

IV. Experimental Evaluation

The five size estimation models given in Table I were applied to the applications in the dataset that was used to derive the models [6]. As baseline models we also estimated the size of the applications in the dataset using the constant and random models.

While carrying out the analysis, we realized that model *ECFP = 7.3 #FPr* is similar to the Average Functional Process (AFP) estimation method proposed by COSMIC [18]. In fact, the AFP estimation model is *ECFP = MSFP×#Fpr*, where *MSFP* is the mean size of functional processes. Therefore, we computed *MSFP* and applied the AFP method as well.

A. The dataset

The dataset we used to evaluate the accuracy of the considered models included data from 23 projects of different nature. More specifically, we used data from 5 projects proposed by the COSMIC consortium to illustrate the counting process, 7 projects from academia, 10 Web-based Management Information Systems (from the same company) and a project management tool. Additional information on the dataset (including the actual data) can be found in [6].

Some descriptive statistics of the dataset are given in Table II.

TABLE II. Descriptive statistics of the dataset

	Size[CFP]	#FPr	AvDGperFPr
Mean	174.1	24.3	4.1
Median	116.0	19.0	3.8
Min	31.0	7.0	2.1
Max	514.0	74.0	8.2
StDev	139.8	16.8	1.5

B. Analysis of errors: Mean Absolute Error

The MARs of the estimates obtained using the models mentioned above are given in Table III, together with the MARs of the constant model (MAR_{CM}) and the random model ($\overline{MAR_{rnd}}$).

In Table III, column rnd 5% gives the value of the 5% quantile of the random estimate MARs. This practice is suggested by Shepperd and MacDonell [11] because the 5% quantile can be interpreted like α for conventional statistical inference, that is, any accuracy value that is better than this threshold has a less than one in twenty chance of being a random occurrence. Therefore, to have reasonable confidence that a given model is actually predicting and not guessing, we expect a value of MAR that is lower than this threshold value. We have that both AFP and the UML-based estimation methods yield MAR values that are well below the proposed threshold, hence we can regard them as not due to chance.

TABLE III. Mean absolute residuals of models.

Name	*Formula*	*MAR*
rnd	–	146
rnd 5%	–	106
CM	–	114
AFP	$ECFP = MSFP\ \#FPr$	52
avg1	$ECFP = 7.3\ \#FPr$	54
reg1	$ECFP = -16.5 + 6.698\ \#FPr$	48
avg2	$ECFP = AvDGperFPr 1.8\ \#FPr$	27
reg2	$ECFP = -64.6 + 7.63\ \#FPr + 9.71\ AvDGperFPr$	40
log2	$ECFP = 1.588\ \#FPr^{1.00357}\ AvDGperFPr^{1.0312}$	25

Note that here we do not explicitly compute SA. Instead, we give the values of MAR needed for the computation. The reason is that with 8 methods there are 28 possible comparison among methods, hence 28 values of SA. Listing all these SA values could create confusion, while to compare two methods' accuracies, we just need to compare their MARs: the model featuring the smaller MAR is likely the best.

C. Distribution of Estimation Errors

In the following sections we shall see that MAR is a quite synthetic indicator, which "hides" important information. To get a deeper insight into estimation errors, in this section the distribution of errors (alias residuals), absolute errors and relative absolute errors are given.

Figure 2 illustrates the boxplots of the errors yielded by each one of the evaluated models. The blue diamonds indicate the mean error for each model. It is interesting to note that AFP and avg1 tend to overestimate, reg1 tends to underestimate, while all the other model neither overestimate nor underestimate.

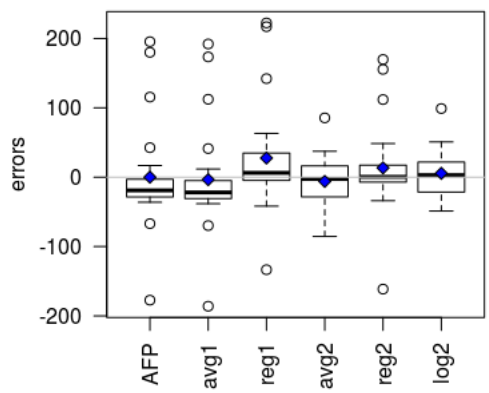


Figure 2. Estimation residuals of the evaluated models.

Figure 3 illustrates the boxplots of the absoluter errors yielded by each one of the evaluated models. The blue diamonds indicate the MARs. Also from this figure we get some interesting result. For instance, the MAR of reg2 is larger than the MARs of avg2 and log2, but the distributions of absolute errors are very similar: the distribution of reg2 is even better than the other two, except for three outliers, which feature absolute errors larger than 150 CFP.

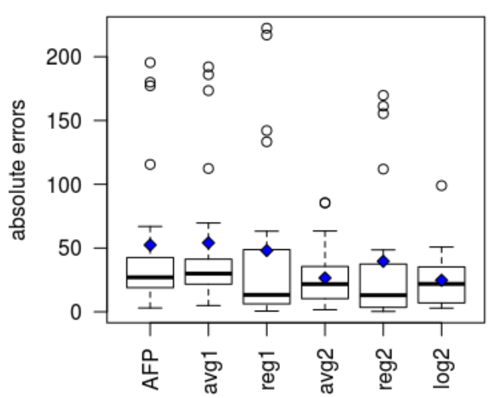


Figure 3. Absolute estimation errors of the evaluated models.

Figure 3 illustrates the boxplots of the relative absolute errors yielded by each one of the evaluated models. The blue diamonds indicate the MMREs.

Relative absolute errors are useful to assess the importance of errors. As a matter of facts, the boxplots show that some models yield some errors that are close to 100%.

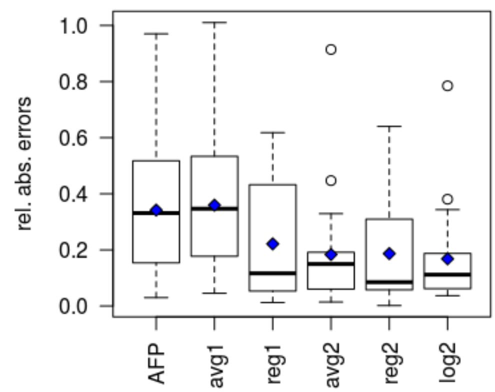


Figure 4. Relative absolute estimation errors of the evaluated models.

D. Analysis of errors: Wilcoxon Signed Rank Test

Table III provides a first piece of evidence: AFP and model-based COSMIC size estimation are definitely more accurate than both the random and constant models.

Table III also confirms that the constant model is more accurate than the random model, as demonstrated by Lavazza and Morasca [13]. For this reason, in the remainder of the paper the random model is no longer used.

To establish if the estimations of one method were significantly better than the estimations provided by another method, we tested the statistical significance of the absolute errors achieved with the two estimation methods [2]. Namely, we compared the absolute residuals provided by every pair of methods via Wilcoxon Sign Rank Test.

The results are given in Table IV, where in each cell the sign "$>$" (respectively, "$<$", "=") indicates that the absolute residuals of the model on the cell's row are larger (resp., smaller, equal) than the absolute residuals of the model on the cell's column.

TABLE IV. COMPARISON OF ABSOLUTE RESIDUALS USING WILCOXON SIGN RANK TEST.

	const	AFP	avg1	reg1	avg2	reg2	log2
const		>	>	>	>	>	>
AFP	<		<	>	>	>	>
avg1	<	>		>	>	>	>
reg1	<	<	<		=	>	>
avg2	<	<	<	=		=	=
reg2	<	<	<	<	=		=
log2	<	<	<	<	=	=	

The results provided by Wilcoxon Sign Rank Test confirm the results given in Section IV-B in that the constant model is less accurate than all other models and AFP is more accurate than avg1 but less accurate than model-based size estimation methods. However, Wilcoxon Sign Rank Test provides further insights with respect to MAR:

- There is no sufficient evidence to conclude that log2 is more accurate than avg2: this fact could be guessed, based on the fact that MAR_{log2} (25) and MAR_{avg2} (27) are quite close.

- Somewhat surprisingly, there is no evidence that either avg2 (which has $MAR_{avg2} = 27$) or log2 (which has $MAR_{log2} = 25$) is more accurate than reg2 (which has $MAR_{reg2} = 40$).
- Similarly, There is no evidence that avg2 (which has $MAR_{reg2} = 27$) is more accurate than reg1 (which has $MAR_{reg1} = 48$).

These results are interesting, in that by just looking at the MAR values we could have concluded that some model (e.g., avg2) is more accurate than another model (e.g., reg2), while –according to Wilcoxon Sign Rank Test– there is no statistically significant evidence of this fact. The explanation of why MAR can be somewhat misleading in this case is given in Figure 5, where the boxplots of the absolute residuals of models avg2 and reg2 are given: it is easy to see that reg2 has a greater MAR because of three applications, whose size estimation error is quite large. Apart from these three applications, the distributions are similar: accordingly, the MARs of avg2 and reg2 are not significantly different.

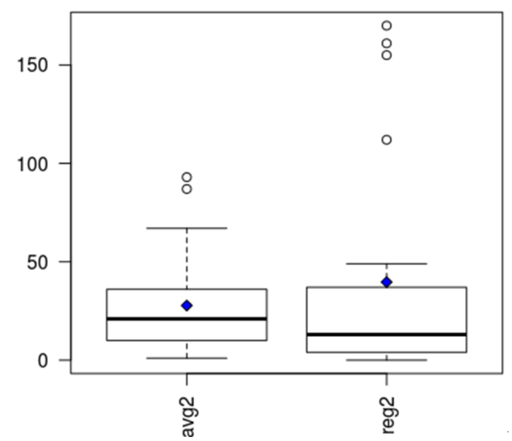


Figure 5. Absolute residuals of models avg2 and reg2.

E. Analysis of errors: Effect Size

Now, as recommended by Shepperd and MacDonell (see Section III-E) we evaluate the effect size. To this end, we computed Hedges's g for all model pairs. The results are given in Table V. where the value for the i^{th} row and j^{th} column is Hedges's g for the pair of models indicated on the i^{th} row and j^{th} column. For instance, the value on row avg1 and column reg1 is 0.09: this indicates that using reg1 instead of avg1 involves a negligible effect.

TABLE V. Effect size (Hedges's g).

	CM	AFP	avg1	reg1	avg2	reg2	log2
CM	–	0.80	0.79	0.82	1.32	0.98	1.36
AFP	-0.80	–	-0.03	0.07	0.55	0.21	0.60
avg1	-0.79	0.03	–	0.09	0.60	0.24	0.64
reg1	-0.82	-0.07	-0.09	–	0.40	0.13	0.44
avg2	-1.32	-0.55	-0.60	-0.40	–	-0.29	0.07
reg2	-0.98	-0.21	-0.24	-0.13	0.29	–	0.34
log2	-1.36	-0.60	-0.64	-0.44	-0.07	-0.34	–

Table V essentially confirms the findings given in the previous sections. It is easy to see that all model-based size estimation methods appear definitely preferable with respect to the constant model. AFP and avg1 appear essentially equivalent. All models involving two independent variables appear better than those based on one independent variable. Models avg2 and log2 appear preferable to the other model-based estimation methods, with log2 only marginally better than avg2.

F. Analysis of errors: IARA

The accuracy of the size estimation models was evaluated via the IARA index described in Section III-F. Specifically, the IARA index was computed for every pair of models. The results are given in Table VI. For instance, the value on row AFP and column reg1 indicates the ratio between the number of estimates for which reg1 was more accurate than AFP and the total number of estimates: 0.62 indicates that reg1 was more accurate than AFP in 62% of the estimates. Although this is a quite straightforward indication, the F in parenthesis indicates that the binomial test fails (i.e., its p-value is not < 0.05), hence the indication is not reliable.

TABLE VI. Comparison of models via IARA indexes

	AFP	avg1	reg1	avg2	reg2	log2
AFP		0.24(F)	0.62(F)	0.67(F)	0.81(S)	0.71(S)
avg1	0.76(S)		0.62(F)	0.71(S)	0.81(S)	0.76(S)
reg1	0.38(F)	0.38(F)		0.52(F)	0.67(F)	0.48(F)
avg2	0.33(F)	0.29(F)	0.48(F)		0.52(F)	0.48(F)
reg2	0.19(F)	0.19(F)	0.33(F)	0.48(F)		0.48(F)
log2	0.29(F)	0.24(F)	0.52(F)	0.52(F)	0.52(F)	

It can be observed that few of the indexes in Table VI are marked "S" for success (i.e., p-value < 0.05). This is largely due to the fact that the dataset used for the evaluation is relatively small. However, it is interesting to note that the IARA index provides indications that are consistent with the Wilcoxon test. In fact, the Wilcoxon signed rank test found that reg2 is not significantly worse than avg2 and log2, although its MAR (40) is larger that avg2 and log2 MARs (27 and 25, respectively). The IARA index tells us that both avg2 and log2 provide more accurate estimates than reg2 only in 52% of cases.

The IARA index appears to add a new dimension to the analysis, complementing the indications (often quite rough) given by the MAR.

V. Discussion of Results

The results of the empirical investigation described in Section IV support consideration concerning a few aspects: the accuracy of model-based methods for estimating the size of software applications expressed in CFP; the practical application of the proposed evaluation methods; the evaluations that can be obtained by directly observing the results yielded by estimation method, without performing statistical analyses.

A. Evaluation of estimation models

With reference to Figure 1, at the end of phase a), we know the number of Functional Processes (#FPr), thus models AFP, avg1 and reg1 are applicable. At the end of phase b), the other models are also applicable.

According to the analysis of experimental data, we have that the models that are applicable at the end of phase b) are –to different extents– more accurate than the models that are applicable at the end of phase a). This was expected, since by progressing from phase a) to phase b), more information concerning the application is made available through UML models, and we can exploit this information to achieve more accurate size estimates. However, having reliable empirical evidence that progressing through application modeling phases enable the construction of progressively more accurate models of the functional size is quite important. It also indicates that collecting measures of COSMIC elements (especially #FPr and #DG, hence AvDGperFPr) and building several statistical models of COSMIC size is useful to get a progressively more accurate notion of the size of the application being built. Actually, the effect size indicators (see Table V) suggest that the models available at the end of phase b) allow only for a medium-small improvement over the best models available at the end of phase a), especially as far as reg1 is concerned. However, to achieve this moderate improvement, all you have to do is counting data groups (i.e., classes in UML models): since this counting is very easy (it can even be automated) building more accurate models at the end of phase b) is not only possible, but most probably always convenient.

B. On the practical application of the proposed indicators

The practical application of the proposed method depends largely on how easily practitioners can derive the indicators mentioned in the sections above. Luckily, the proposed technique are supported by open-source tools, and can be automated quite easily. Specifically, the language and programming environment R [19] supports the computation of all of the proposed indicators; you just need to install the 'effsize' package [20] to compute Hedges's g.

So, once the actual and estimated values are available, computing the indicators described in Section III (as well as plots given in Section V-C below) can be completely automatic, and quite fast.

In conclusion, the proposed indicators are easily obtained and are also fairly easy to interpret; therefore, practitioners should have no difficulty to assess the accuracy of estimation methods as proposed.

C. Direct observation of residuals

The proposed approach is statistically sound and appears to provide reliable indications. Moreover, as discussed in Section V-B, it can be automated. Nonetheless, one should not forget that in general the direct observation of the estimates and their comparison with actuals can provide quite enlightening indications.

For instance, suppose that you are interested in evaluating the accuracy of the avg2 and log2 models. To this end, you could look at a plot like the one in Figure 6. In Figure 6, the x axis reports the actual size of applications (in CFP), and the y axis reports the estimated size (in CFP); applications appear as small circles, estimates obtained using avg2 appear as green crosses, and estimates obtained via log2 appear as red xs.

By looking at Figure 6, appreciating the accuracy of the estimation models is easy. It is also easy to compare the estimates yielded by avg2 and log2. For instance, it can be noticed that log2 estimates are always smaller than avg2 estimates: this observation may be very relevant for a practitioner who has to choose whether to use avg2 or log2.

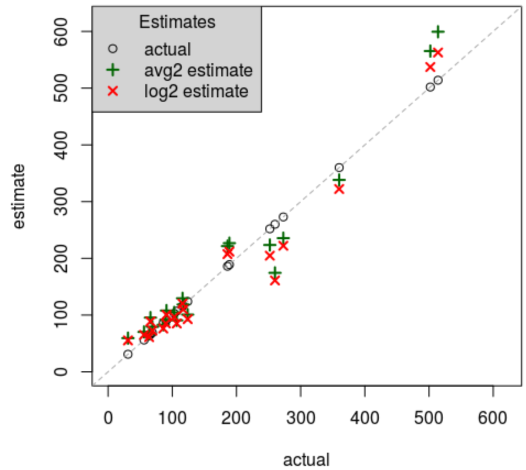


Figure 6. avg2 and log2 estimates vs. actuals.

As is often the case, there are many relatively small applications. In our case, it is difficult to assess the estimates of applications in the [50,150] CFP range. To overcome this difficulty, you can draw another plot that accounts for the applications smaller than 150 CFP only. Such plot is shown in Figure 7: the plot confirms that log2 estimates are always smaller than avg2 estimates also for relatively small applications.

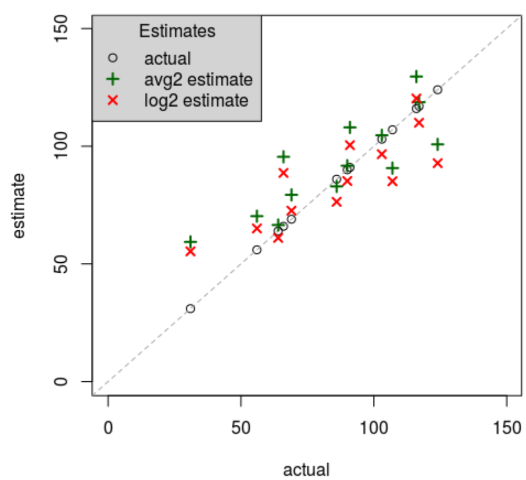


Figure 7. avg2 and log2 estimates vs. actuals (detail on applications smaller than 150 CFP).

VI. Threats to Validity

Like in any empirical study, we have to deal with some threats to the validity of our analysis.

We see no construction issues with our analysis, since all the used techniques are statistically sound; in fact, they have been proposed to correct the problems with previous indicators, such as MMRE.

The main problem we face is probably the generalizability of results. In fact, our results derive from the analysis of a dataset that collects data from only 21 applications. It is possible that other datasets could support somewhat different conclusions. However, the fact that our dataset includes several industrial applications, and that the size of the dataset is not excessively small (especially in the context of empirical software engineering studies) supports the hypothesis the results presented here are sufficiently representative in general. Also, the logical coherence of the results –namely the fact that the more information is available from UML models, the more accurate is size estimation– supports the hypothesis the results presented here are valid.

VII. Related Work

The accuracy evaluation techniques used in this paper are being increasingly used by researchers that need to evaluate the accuracy of new effort estimation proposals. For instance, Sarro et al. used the Mean Absolute Error and the Standardized Accuracy to assess the accuracy of a bi-objective effort estimation algorithm that combines confidence interval analysis and assessment of mean absolute error [21]. To establish if the estimations of one method were significantly better than the estimations provided by another method, they tested the statistical significance of the absolute errors achieved with different estimation methods via the Wilcoxon Signed Rank Test, as we did in Section IV.

The techniques used here are becoming quite popular, but there are also several alternative proposal, actually too many to be mentioned here. As an example of an alternative to SA, Tofallis proposed to use the logarithm of the accuracy ratio: $log\frac{prediction}{actual}$ [22]. As an example of an alternative to Hedges's g, Vargha and Delaney proposed the A12 statistic, a non-parametric effect size measure: given a performance measure M, A12 indicates the probability that running algorithm A yields higher M values than running another algorithm B [23]. Finally, a quite different but interesting proposal is StatREC, a Graphical User Interface statistical toolkit designed to provide a variety of graphical tools and statistical hypothesis tests to facilitate strategies for an intelligent decision-making [24].

Concerning the assessment of accuracy of functional size estimation methods, to the best of the author's knowledge, very little work has been done. In general, some evaluation is done when a method is proposed, as in [25], where the NESMA estimated method is proposed and its accuracy is evaluated on the training set. A noticeable exception is [26], where several early estimation methods for Function Point measures are evaluated via an empirical study.

The method used in this paper was partly applied to evaluate the accuracy of measure conversions (from IFPUG Function Points to COSMIC Function Points) in [27].

VIII. Conclusion

In this paper, the accuracy of a set of model-based methods to estimate the COSMIC size of software applications has been evaluated. The relevance of the paper is based on two observations.

First, practitioners are very keen to know the accuracy that can be achieved via size estimation methods. In fact, they often use size estimation methods to derive the most important piece of information upon which the cost of software is estimated; therefore, accurate size estimation is essential to get accurate cost estimates, hence to allocate the correct amount of resources and prepare reliable development plans.

Second, to evaluate the accuracy of estimates, you need reliable indicators. Traditional indicators like MMRE have been proved to be biased: thus, finding and testing more reliable indicators is necessary. Consider for instance what happens when researchers propose a new estimation technique: how can we decide that the new technique is good, and possibly even better than existing techniques? Reliable accuracy evaluation techniques and indicators –like those proposed here– can answer such question.

According to our empirical study, we can recommend that the accuracy of estimates be evaluated by

- Computing the mean of absolute residuals (MAR) of all the models to be tested.
- Any estimation method must prove more accurate than the trivial models –like the constant model and the random model– that do not require any knowledge concerning the application to be estimated. Hence, one should always test models against the random and constant models.
- Having established that the considered estimation method is better than the trivial methods, one should also evaluate whether the considered method is more accurate than the currently used estimation technique, to see if abandoning the latter to adopt the new method is worthwhile.
- Using Wilcoxon Sign Rank Test is advisable, since it can give indications concerning the statistically significance of the comparisons based on two methods' MAR values.
- Also looking at the boxplots of absolute residuals can help, especially when a few outliers affect the MAR at a great extent (as in Figure 2).
- Assessing the effect size using Hedges's g (or similar indicators) is useful to assess the extent of the improvement that a new technique can guarantee over another one.
- Finally, the IARA index shows in how many cases a method is more accurate than another one.

Overall, this paper shows that assessing the accuracy of estimates can hardly be based on a single indicator. Instead, using the proposed set of indicators provides a quite detailed and complete picture of the merits of the evaluated estimation models.

As a final observation, we note that the analyses reported in this paper were carried out in the R environment [19]. Practitioner and researchers that need to evaluate estimation accuracy can apply the proposed approach quite easily and derive the indicators described in Section III very quickly.

Future work includes:

- Further evaluating model-based COSMIC size estimation methods against additional datasets.
- Experimenting the accuracy evaluation methods used in this paper with other estimation techniques and using other datasets.

Acknowledgment

The work presented here has been partly supported by the "Fondo di Ricerca d'Ateneo" of the Università degli Studi dell'Insubria.

References

[1] L. Lavazza, "Accuracy Evaluation of Model-based COSMIC Functional Size Estimation," in International Conference on Software Engineering Advances (ICSEA), 2017.

[2] B. A. Kitchenham, L. M. Pickard, S. G. MacDonell, and M. J. Shepperd, "What accuracy statistics really measure," IEE Proceedings-Software, vol. 148, no. 3, 2001, pp. 81–85.

[3] T. Foss, E. Stensrud, B. Kitchenham, and I. Myrtveit, "A simulation study of the model evaluation criterion MMRE," IEEE Transactions on Software Engineering, vol. 29, no. 11, 2003, pp. 985–995.

[4] I. Myrtveit, E. Stensrud, and M. Shepperd, "Reliability and validity in comparative studies of software prediction models," IEEE Transactions on Software Engineering, vol. 31, no. 5, 2005, pp. 380–391.

[5] The COSMIC consortium, Functional Size Measurement Method Version 4.0.1 Measurement Manual (The COSMIC Implementation Guide for ISO/IEC 19761:2011), 2015.

[6] V. Del Bianco, L. Lavazza, G. Liu, S. Morasca, and A. Z. Abualkishik, "Model-based early and rapid estimation of COSMIC functional size– an experimental evaluation," Information and Software Technology, vol. 56, no. 10, 2014, pp. 1253–1267.

[7] K. Berg, T. Dekkers, and R. Oudshoorn, "Functional size measurement applied to UML-based user requirements," 2005, pp. 69–80.

[8] L. A. Lavazza, V. Del Bianco, and C. Garavaglia, "Model-based functional size measurement," in Proceedings of the Second ACM-IEEE international symposium on Empirical software engineering and measurement. ACM, 2008, pp. 100–109.

[9] A. Živkovič, I. Rozman, and M. Heričko, "Automated software size estimation based on function points using UML models," Information and Software Technology, vol. 47, no. 13, 2005, pp. 881–890.

[10] L. Lavazza and V. Del Bianco, "A case study in COSMIC functional size measurement: The rice cooker revisited," Software Process and Product Measurement, 2009, pp. 101–121.

[11] M. Shepperd and S. MacDonell, "Evaluating prediction systems in software project estimation," Information and Software Technology, vol. 54, no. 8, 2012, pp. 820–827.

[12] W. B. Langdon, J. Dolado, F. Sarro, and M. Harman, "Exact mean absolute error of baseline predictor, MARP0," Information and Software Technology, vol. 73, 2016, pp. 16–18.

[13] L. Lavazza and S. Morasca, "On the evaluation of effort estimation models," in Proceedings of the 21st International Conference on Evaluation and Assessment in Software Engineering. ACM, 2017, pp. 41–50.

[14] J. Cohen, Statistical power analysis for the behavioral sciences. Hillsdale, NJ: Lawrence Earlbaum Associates, 1988.

[15] R. Rosenthal, H. Cooper, and L. Hedges, "Parametric measures of effect size," The handbook of research synthesis, 1994, pp. 231–244.

[16] P. D. Ellis, The essential guide to effect sizes: Statistical power, meta-analysis, and the interpretation of research results. Cambridge University Press, 2010.

[17] J. Cohen, "A power primer." Psychological bulletin, vol. 112, no. 1, 1992, pp. 155–159.

[18] F. Vogelezang, C. Symons, A. Lesterhuis, R. Meli, and M. Daneva, "Guideline for Early or Rapid COSMIC Functional Size Measurement by using approximation approaches," July 2015.

[19] R Core Team, R: A language and environment for statistical computing. R Foundation for Statistical Computing, 2014.

[20] M. Torchiano, "Efficient Effect Size Computation – package 'effsize'," March 2017. [Online]. Available: https://cran.r-project.org/web/packages/effsize/effsize.pdf

[21] F. Sarro, A. Petrozziello, and M. Harman, "Multi-objective software effort estimation," in Proceedings of the 38th International Conference on Software Engineering. ACM, 2016, pp. 619–630.

[22] C. Tofallis, "A better measure of relative prediction accuracy for model selection and model estimation," Journal of the Operational Research Society, vol. 66, no. 8, 2015, pp. 1352–1362.

[23] A. Vargha and H. D. Delaney, "A critique and improvement of the cl common language effect size statistics of mcgraw and wong," Journal of Educational and Behavioral Statistics, vol. 25, no. 2, 2000, pp. 101–132.

[24] N. Mittas, I. Mamalikidis, and L. Angelis, "A framework for comparing multiple cost estimation methods using an automated visualization toolkit," Information and Software Technology, vol. 57, 2015, pp. 310–328.

[25] H. van Heeringen, E. van Gorp, and T. Prins, "Functional size measurement-accuracy versus costs-is it really worth it?" in Software Measurement European Forum (SMEF), 2009.

[26] L. Lavazza and G. Liu, "An empirical evaluation of simplified function point measurement processes," International Journal on Advances in Software, vol. 6, no. 1 & 2, 2013, pp. 1–13.

[27] A. Z. Abualkishik and L. Lavazza, "IFPUG Function Points to COSMIC Function Points convertibility: A fine-grained statistical approach," Information and Software Technology, vol. 97, 2018, pp. 179–191.

HyDE: A Hyper-Display Environment in Mixed and Virtual Reality and its Application in a Software Development Case Study

Roy Oberhauser, Alexandre Matic, and Camil Pogolski
Computer Science Dept.
Aalen University
Aalen, Germany
email: {roy.oberhauser, camil.pogolski}@hs-aalen.de, alexandre.matic@studmail.hs-aalen.de

***Abstract* - While Virtual Reality (VR) has been applied to various domains to provide new visualization and interaction capabilities, enabling programmers to utilize VR for their software development and maintenance tasks has been insufficiently explored. In this paper, we present the Hyper-Display Environment (HyDE) in the form of a mixed-reality (HyDE-MR) or virtual reality (HyDE-VR) variant respectively, which provides simultaneous multiple operating system window visualization with integrated keyboard/mouse viewing and interaction using MR or in pure VR via a virtual keyboard. This paper applies HyDE in a software development case study as an alternative to typical non-VR Integrated Development Environments (IDEs), supporting software engineering tasks with multiple live screens in VR as an augmented virtuality. The MR solution concept enables programmers to benefit from VR visualization and virtually unlimited information displays while supporting their more natural keyboard interaction for basic code-centric tasks. Thus, developers can leverage VR paradigms and capabilities while directly interacting with their favorite tools to develop and maintain program code. A prototype implementation is described, with a case study demonstrating its feasibility and an initial empirical study showing its potential.**

Keywords* - *virtual reality; mixed reality; augmented virtuality; integrated development environments; software engineering; computer-aided software engineering; programming.

I. INTRODUCTION

This paper is an extension of [1], which described our Mixed-Reality Fly-Thru-Code (MR-FTC) approach for visualizing software structures in virtual reality (VR) and supported coding via a virtual tablet and integration of a mixed-reality (MR) keyboard.

As digitalization sweeps across society, the amount of program source code created and maintained worldwide is steadily increasing. Google is said to have at least 2bn lines of code (LOC) accessed by over 25K developers [2], while GitHub has over 79m repositories and 28m developers [3]. It has been estimated that well over a trillion LOC exist with 33bn added annually [4]. This is exacerbated by a limited supply of programmers and high employee turnover rates for software companies, e.g., 1.1 years at Google [5]. This has ramifications on the labor expenses involved in software development and maintenance. Approximately 75% of technical software workers are estimated to be doing maintenance [6]. Moreover, program comprehension may consume up to 70% of the software engineering (SE) effort [7]. As an example, the Year 2000 (Y2K) crisis [8] with global costs of $375-750 billion provided an indicator of the scale and importance of program comprehension. Activities involving program comprehension include investigating functionality, internal structures, dependencies, run-time interactions, execution patterns, and program utilization; adding or modifying functionality; assessing the design quality; and domain understanding of the system [9].

Some of the challenges faced by software developers who are now more than ever often facing unfamiliar preexisting codebases are: 1) effectively and efficiently familiarizing and comprehending the structure and intent of collaboratively developed code, 2) programming and testing code changes, and 3) maintaining (debugging, optimizing, or securing) the code. Yet the tools programmers use to work with this code have not significantly changed over the years. The Jolt Productivity Award for 2015 went to an IDE, the Jetbrains IntelliJ IDEA, and other IDEs, Apple's Xcode and Microsoft's Visual Studio, were finalists [10]. RebelLabs Developer Productivity Report asked what tools developers most used and 3 IDEs were reported (IDEA, Eclipse, NetBeans) [11].

Within the scope of developer's program comprehension and informational challenges, one could hypothesize that the simultaneous access to situationally relevant information by developers is at least in part hampered by current physical limitations for viewability on computer displays. This can be observed by the - not infrequent - use of multiple displays (when space and budget allow), high resolutions (when visibility allows), and the multiple windows and tabs open by developers during their tasks, where quick access to relevant information is critical.

As to possible visual interface solutions, a survey of over 21K developers by SlashData in 2017 showed that 25% of professional game developers were targeting VR or MR headsets [12]. That indicates that a growing segment of software developers are becoming familiar with and have access to these headsets during development, whereby the target environment for which the headsets are intended are gamers and not developers. Thus, it may be viable to instead leverage the opportunities afforded by VR [13] and MR interfaces to support and target software developers during their development and provide more comprehensive information. However, a peculiarity and challenge regarding using VR with software developers in contrast to typical VR users is their affinity to keyboard interfaces when interacting with program code. This would thus typically require frequent (un)fastening of the VR headset to view and utilize the real keyboard - an annoying disruption, visually disconcerting from the immersion experience, and inefficient. Alternatively, virtual keyboards would require selecting one key at a time using the typical VR controllers (akin to one finger typing) or some unfamiliar spin-dial approach and would be an inefficient means for code input for various programmers, since they often require special characters. While data gloves might become a future interface alternative for typing, they have not yet become sufficiently popular.

In our prior work we described our Mixed-Reality Fly-Thru-Code (MR-FTC) approach [1], which provided an immersive software structure visualization and fly-thru experience of code structure dependencies and enabled programmers to view code and make text changes on a virtual tablet using MR for keyboard and mouse access. Here we extend [1] by removing the limitations of the virtual tablet and focusing on enhancing the informational display capabilities in the VR environment with a multi-display and heterogeneous tool source capability while hiding the VR software structure visualization.

This paper contributes the Hyper-Display Environment (HyDE) solution concept, and applies it in a case study to enhance available software developer environment capabilities by leveraging VR to integrate multiple information screens in support of software development and maintenance tasks. As visual IDEs often contain multiple sub-windows with information, HyDE displays an unlimited number of operating system (OS) windows that can contain any tool or information desired. Thus, direct integration support of various IDEs, tools, and information sources is enabled as a type of MR projection into the VR environment. The HyDE-MR variant also integrates real keyboard and mouse/trackpad; the HyDE-VR variant provides a virtual keyboard. The solution concept is sufficiently general to be applicable to any domain desiring simultaneous access to multiple informational screens.

The paper is organized as follows: the next section discusses related work; Section III then describes the solution concept. Section IV provides details about our prototype implementation of the solution concept. In Section V, the evaluation, based on a case study, is described, which is followed by a conclusion.

II. Related Work

Work related to viewing desktop applications in virtual or immersive environments or using hyper displays includes VEWL [14], a library for developing applications projecting windows onto polygons within an immersive virtual environment and provide additional information and controls including menus, windows, and buttons. The user's head and a wand are tracked. CAVE2 [15] is a cylindrical system of 72 passive stereo LCD panels that provide a 320-degree panoramic environment for displaying information, either dedicated to one virtual simulation or having a traditional tiled display wall enabling users to work with large numbers of documents at the same time.

Work related to improving IDEs for SE includes IDE++ [16] is an IDE extension framework and interaction monitor and describes four applications (DevTime, Sage, Proctor, and Localizer) with the intent to make IDEs more intelligent. Code Bubbles [17] attempts to improve the IDE user interface with lightweight editable fragments of code using a bubble metaphor.

VR-related work in support of SE tasks includes software visualization such as Imsovision [18], which visualizes object-oriented software in VR using electromagnetic sensors attached to shutter glasses and a wand for interaction. ExplorViz [19] is a JavaScript-based web application that uses WebVR to support VR exploration of 3D software cities using Oculus Rift together with Microsoft Kinect for gesture recognition. These approaches lack the integration of a keyboard and are thus limited in their ability to support programming.

With regard to MR and augmented reality (AR) support for programming tasks, Tangible Windows [20] provides one open window per tablet and allows the user to switch their application by switching between tablets. Lee et al. [21] describe an approach for authoring tangible augmented reality applications with regard to scenes and object behaviors within the AR application being built, so that the development and testing of the application can be done concurrently and intuitively throughout the development process. However, integration of AR support for non-AR software development is not shown. Billinghurst and Kato [22] show possible concepts for collaboration in VR, but do not depict a keyboard or show how programming task support would work. Neumann et al. [23] do not appear to use VR goggles in augmented reality (AR) for projecting multiple PC screens. In 3d live [24], users view a two-dimensional fiducial marker using a video-see-through augmented reality (AR) interface. Kato and Billinghurst [25] use optical see-through MR, whereby an AR conferencing system was developed that allowed virtual images of remote collaborators to be overlaid on multiple users' real environments. Gupta et al. [26] use a tracking framework, wherein the 3D position of planar pages is monitored as they are turned back and forth by a user, and data is correctly warped and projected onto each page at interactive rates. In each frame, feature points are independently extracted from the camera and projector images and matched in order to recover the geometry of the pages in motion. The book can be loaded with multimedia content, including images, videos, and volumetric datasets.

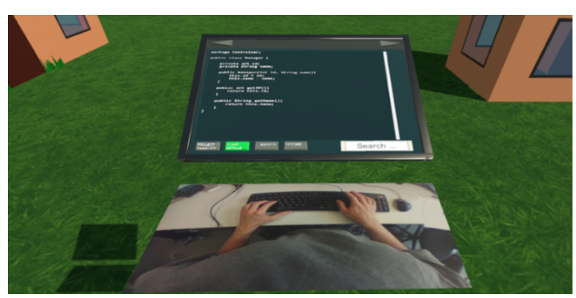


Figure 1. Coding with MR view of keyboard and mouse blended in and scroll bar shown on the virtual tablet.

In our prior MR-FTC approach [1], we visualized software structures using various metaphors and utilized MR with a virtual tablet (called the oracle) to view and edit code for programming tasks, as shown in Figure 1. However, it was limited in functionality and it was not possible to integrate and access other heterogeneous tools.

In contrast to the above work, the HyDE approach leverages VR to enhance simultaneous informational display viewing and interaction, while also supporting basic programming and command-line interface tasks by integrating via MR keyboard and mouse viewing and display interaction in the VR environment.

III. Solution

Our solution concept is domain independent and can be applied to various domains. The information screens displayed are the projections of actual windows from the real operating system and can thus be seen as a form of MR that mixes into the VR environment actual window screens, or more precisely augmented virtuality since the model is mostly virtual and only a relatively small portion is reality. We apply our solution here to the SE or software development area because of its challenges to highlight the solution's potential and capabilities. For instance, to support software developers within the VR environment, our solution concept has an MR variant for integrating keyboard and mouse access which can provide enhanced Graphical User Interface (GUI) and textual interaction for program coding task support.

A. Conceptual Architecture

The GUI from a PC environment can be incorporated in the VR environment projected into the VR landscape of VR goggles or a VR screen (e.g., Google Cardboard) of a VR user, providing a form of MR. As shown in , the operating system (OS) environment (e.g., Windows, Linux, or MacOS) contains various windows or the entire screen of running processes or applications that are viewable ((A-F)).

The VR Environment can incorporate any number of virtual displays of any size placed anywhere in VR space (e.g., a single gigantic virtual display showing all screens (A-F) as separate windows on it) (n-to-1 relationship); correlating screens (1-to-1 relationship); or additional screens that perhaps show historical content or duplicated content (n-to-m relationship). This permits a user to have access to many more screens than physically feasible. Furthermore, various computers limit the number of physical displays that can be attached.

The mechanism to integrate the content of some subset of these windows (or full screen) utilizes available screen capture mechanisms of the operating systems accessed from within the VR Game Engine ((3)). Screenshots are then represented in the VR environment as a texture that can be placed and updated/refreshed (6) on any game object surface, such as one that looks like a virtual display (A-F in the VR Environment Display). These screenshot sequences represent a stream (capture stream) and can be thought to be equivalent to a video stream. A historical view of older screen point in time can be displayed by storing the screenshots and retrieving them (5), permitting time lapse or pausing of screen content. A separate optional Screen Capture Server can be used to set the active window to the foreground and capture the screen (1), and then return it to the background and pass the captured screen image in a place accessible to the VR Game Engine (2).

Interaction with the screens in the VR Environment Display can be supported to effect changes in the PC Environment by eliciting events (7) such as keyboard or mouse events via Input Device mechanisms (e.g., a VR controller, (virtual) keyboard, (virtual) mouse) that can then be transformed and passed on to the OS window (8) as OS events. By utilizing remote desktop applications, one can also view or interact with content across various remote operating system GUIs or windows from within the VR environment.

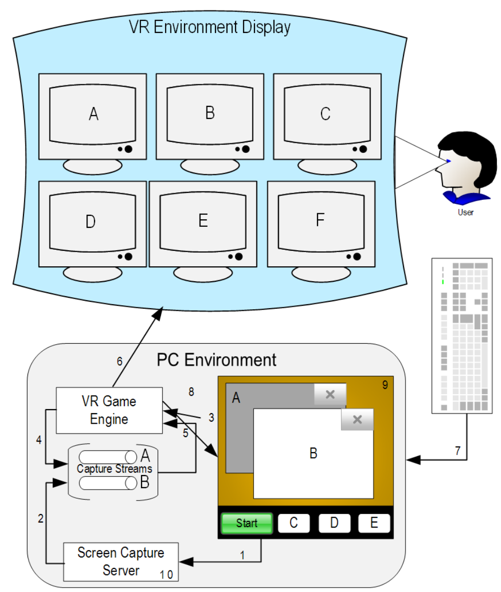


Figure 2. HyDE conceptual architecture.

For the MR variant, a live camera view of the keyboard and mouse can be integrated into the VR landscape. This allows the user to determine where their hands and fingers are relative to the actual hardware. Assuming the subject is seated, for instance, tilting their head down is interpreted as a gesture to activate a live webcam on the VR headset, activating MR mode, similar to the natural head movement made on a desktop PC to look at the keyboard.

B. Process

As shown in Figure 3, the process used by the solution approach can involve the following steps.

For the Window Capture Server:

1) Continually iterate over all processes with GUI windows.

2) Place the window in the foreground.

3) Capture an image of the window using the OS. This can alternatively be an image capture of the screen and then cropped to the foreground window. For the special case of the desktop, the entire screen is captured.

4) Place the image in the image capture stream queue. This can also be persisted if historical records are desired.

5) Return the foreground window to the background.

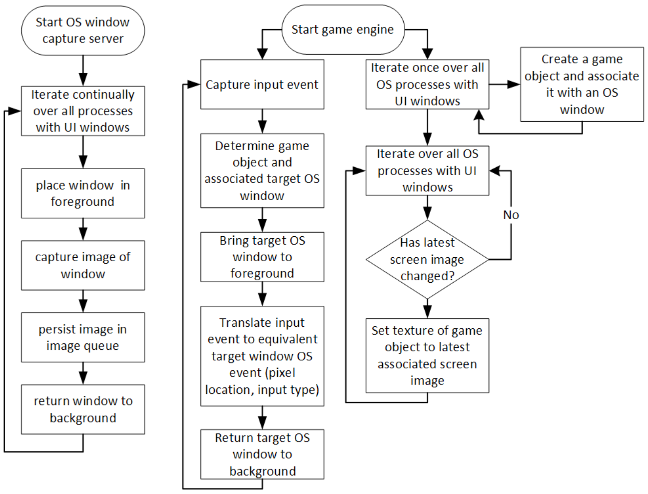


Figure 3. HyDE process steps.

The steps involved in the game engine process for displaying the screen information are as follows:

1) To create a display in VR, a game object is created and associated to a selected OS window process.

2) In a continual iteration, the texture of any display (a game object) is replaced by the image from that window's capture stream (equivalent to a refresh). As a potential optimization, if no change to the image is detected, no update is required. If the stream is paused, no refresh is invoked. On continuation, either time lapse display (historical playback) or current (discarding all images except the latest) are possible. From a persisted capture stream, any timestamp available can be displayed. This image is overlaid with the mouse pointer and cursor position.

The steps involved in the game engine process to support screen interaction are as follows:

1) For the active display, the associated target OS window is determined

2) The window is brought to the foreground.

3) The input event (mouse or keyboard) is applied to this window

4) The window is returned to the background.

The ongoing capturing process takes care of updating the screen for any resulting changes.

IV. Implementation

A prototype was implemented to determine the feasibility of the solution approach. The Unity game engine 2017.3.0b9 was utilized for the visualization due to its multi-platform support, VR integration, and popularity. Blender 2.79 was used to develop all models. For VR hardware we used HTC Vive, a room scale VR set with a head-mounted display with an integrated camera and two wireless handheld controllers tracked using two 'Lighthouse' base stations.

A. Mixed Reality Variant

For MR, we integrated a live camera view into the VR landscape via a virtual plane object. For a better picture, a Logitech C920 webcam with a 1080p resolution was used instead of the Vive Front camera and a backlit keyboard Corsair K70 RGB Lux. Figure 4 shows the MR setup.

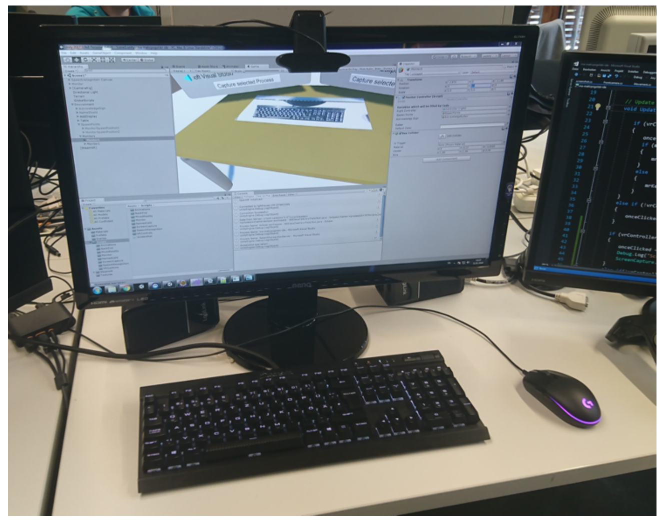


Figure 4. MR setup.

As shown in Figure 5, this allows the user to determine where their hands and fingers are relative to the actual keyboard.

Figure 5. MR, showing real keyboard in cutout of VR desk and displays.

Figure 6 shows a close-up view as to the readability of the keys on the keyboard, with a part of the monitor shown at the top (the webcam view could of course be adjusted to not show the monitor). One advantage of the MR view versus the VR keyboard variant is that the user can utilize their favorite keyboard and mouse that they are already accustomed to.

Figure 6. HyDE-MR variant closeup view of the keyboard.

To avoid distraction, it can be configured to hide the MR view as follows: when the VR user tilts their head down, it is interpreted as a gesture to activate the live webcam on the VR headset and blend this into the virtual plane object. When the head is tilted starkly up, MR is deactivated.

B. *Virtual Reality Variant*

As shown in Figure 7, the HyDE-VR variant avoids MR, requiring the selection of all key and mouse control inputs on any screen be done via the VR controllers.

Figure 7. HyDE-VR variant showing a VR (instead of the MR) keyboard.

While data gloves might provide a better option for typing in VR, they are not yet in widespread use within the VR community. We thus opted at this time for practical variants, the MR utilizing any already available keyboard and mouse or the pure VR variant relying on VR controllers only.

C. *Display Placement and Interaction*

The left controller touchpad allows one to move forwards and backwards (like zoom in or out relative to the displays) and left and right. The right controller touchpad allows one to move up or down. The left controller pointer and trigger can be used to select a display and move it to another position (swap). By pointing and triggering with the right controller on a display, one can freeze (unfreeze) it if one wishes to pause (continue) the capturing (see Figure 8).

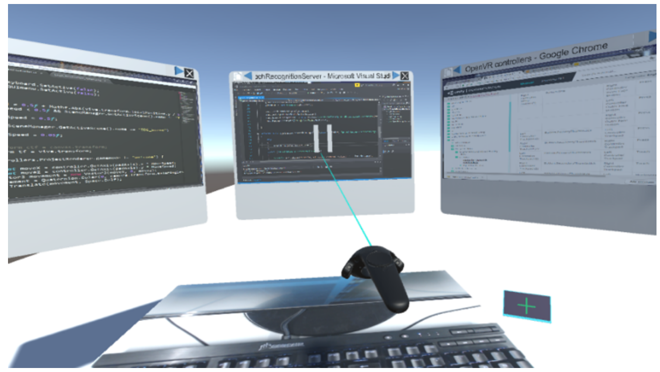


Figure 8. HyDE-MR variant showing pause of capture stream.

New displays are created by pressing the '+' button near the keyboard (see Figure 6 or Figure 7) using the VR controller laser pointer (seen in Figure 9). Placement of the displays is as follows: the first display is placed in a fixed position in front of the keyboard. Further displays are placed relative to the first, the second and third are placed to the left and right of the first at a 45° vertical angle. The fourth through sixth are placed on a second higher row and tilted forward for a better viewing angle without having to move the VR camera position. A bottom third row is placed for the seventh through ninth in a tilted backward manner.

Figure 9. Developer MR desktop setup, with Visual Studio shown on left virtual display, and Eclipse IDE on right virtual display.

Once the tenth display is reached, a stacking configuration of rows of three displays is applied with all rows stacked vertically (no tilting) and the left and right columns angled at 45° without tilting (see Figure 10). By moving relative to this stack the view is adjusted to bring the display of interest up or down and in front to the best viewing angle.

Arrows along the top edge allow one to scroll through the circular list of processes to pick the one to show on that display (see Figure 11). The one to pick is done by pointing the right controller at the process name text and selecting it with the trigger (the process name scrolls when the string is too long for the available space).

As shown in Figure 12, the displays can be closed via an 'x' button on their upper right corner.

Figure 10. HyDE display stacking.

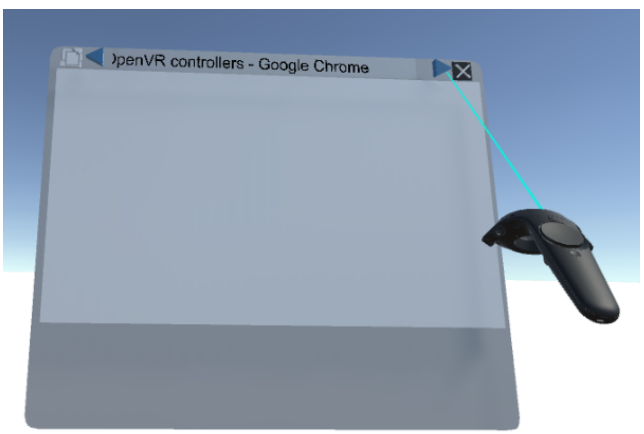


Figure 11. Selecting a process name.

Figure 12. Closeup of a live IDE on virtual display.

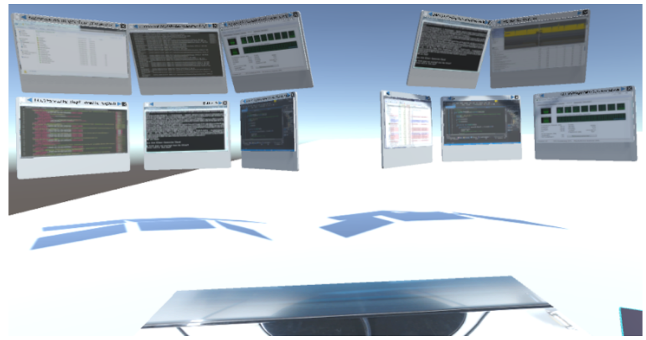


Figure 13. HyDE situationally-related multi-area concept.

A new Area (a group of situationally-relevant displays) can be created by moving sufficiently away from the current Area and pressing the create display button. Figure 13 shows two areas. For example, one area could be focused on fixing a bug from a previous release, while another is focused on optimizing a performance issue, or another is focused on developing some new feature

In case the desired process is not yet running, a special 'null' process can be selected, which will display the desktop so the user can start the desired application, at which point the user can select the process name.

D. *Implementation Details*

The Server is programmed in C# using .NET 4.7 and runs on Microsoft Windows. The Server holds all pertinent information about each running process with an open window.

Unity acts as a client and binds via TCP-IP with the Server to retrieve all process information in XML. Information provided are: process name, process ID, coordinates of the top left position of the open window, window width and height (in pixels), and the window handle. If a process name is selected, the window is captured. If a window is moved, the handle can be used to determine its new position.

Capturing of a process window is done by making the process window the active one (in the foreground) and then capturing the entire screen and then this image is then cropped to the known coordinates of the desired window. This is done to support displaying the cursor. Using a variable of type CURSERINFO, information about the cursor position can be retrieved and its position drawn as an Icon and overlaid on the cropped window picture when a cursor is in that window. Each captured window is handled by a separate thread. For each window the mouse cursor is tracked so it displays the appropriate position for that screen.

V. EVALUATION

The evaluation of the HyDE solution concept consists of a case study and empirical study in software development.

A. *Case Study*

In this case study, we take a typical SE maintenance situation where a software developer is attempting to address a bug report that is related to a distributed application. Note that we exemplify this fictional case with analogous screens that do not directly relate to each other nor show actual bug information, they are however from actual live screens.

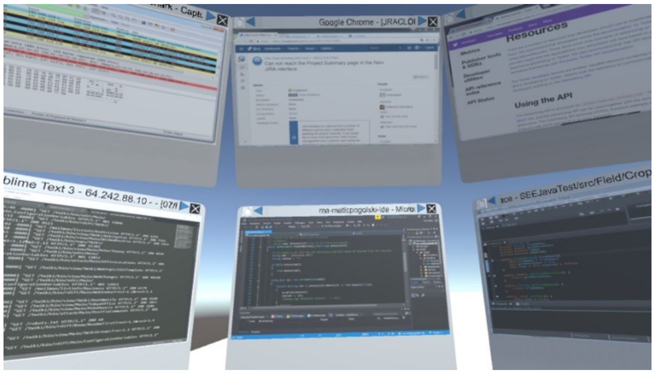


Figure 14. Closeup of a live IDE on virtual display.

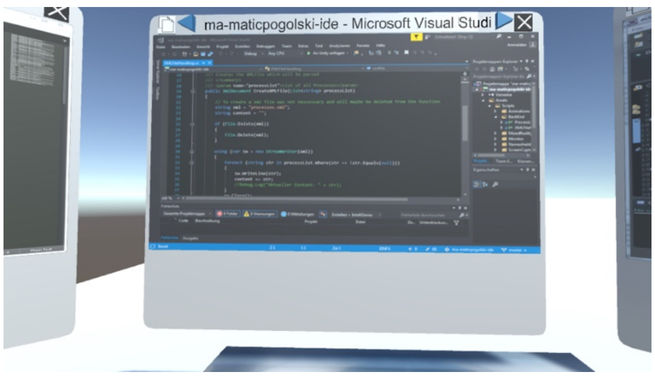


Figure 15. Closeup of a live Visual Studio IDE on virtual display.

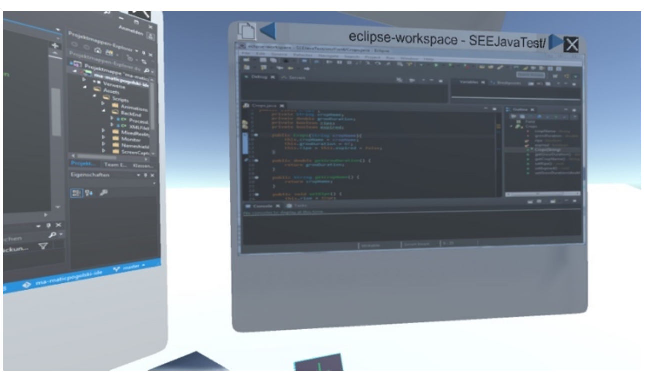


Figure 16. Closeup of a live Eclipse IDE on virtual display.

Figure 14 shows the bug report information in the Google Chrome web browser in the center display of the upper row. Perhaps it is unclear if the .NET client is not making a network-based Representational State Transfer (REST) call to the backend Java-based service, or if the service did not return. For this, related information might be to analyze certain log files to see if there are any warnings or errors. Figure 14 shows the bottom left screen with a log file opened in the Sublime Text editor. The client may have been written in a Common Language Infrastructure (CLI) language and for which Visual Studio might be an applicable IDE, shown in the bottom center display. The documentation for the backend service web application programming interface (API) may need to be consulted to familiarize oneself with the call and parameters specifications, shown in the upper right display. Debugging of the backend service, which is written in Java, is done using another IDE, in this case Eclipse shown on the bottom right display. To actually monitor the network packets, the Wireshark tool might be used, shown in the upper left display.

Figure 15 shows a close-up of the Visual Studio IDE, while Figure 16 shows a close-up of the Eclipse IDE.

B. *Empirical Evaluation*

For an empirical evaluation, for our experiment design we chose to compare subject performance for program debugging tasks using HyDE-MR. The interface type (VR-based HyDE vs. a non-VR notebook) is an independent variable and effectiveness and efficiency are dependent variables. A convenience sample of seven Computer Science (CS) students and one Information Systems (IS) student, who were either in their senior year or master students, was selected with only one unfamiliar with VR. Only one indicated not personally using a multi-monitor setup.

The experiment was supervised and a brief training to show how to utilize HyDE functionality was initially given. Due to only having access to one (heavily shared) VR station and subject constraints, we chose to create an SE task set that required a maximum of 60 minutes per subject (30 minutes maximum in VR) and was basic enough that students across various semesters could do it. We intentionally injected 8 errors into a webpage consisting of HTML, Cascading Style Sheets (CSS), and JavaScript errors (e.g., text that indicates it should be centered but is not). The intended functionality was documented directly in the webpage and it was up to the subject to analyze the text and determine if the webpage was functioning properly for that requirement, and if not, to indicate that a defect was found and then correct it. One set of errors were made for normal monitor (non-VR) usage scenario (see Figure 17), and another set for the VR usage scenario (see Figure 18). After correction, the non-VR webpage should like Figure 19 while the VR webpage should like Figure 20. In the non-VR scenario, the subjects had access to one display but could use multiple windows. In the VR scenario, we observed them creating and using 4-5 displays.

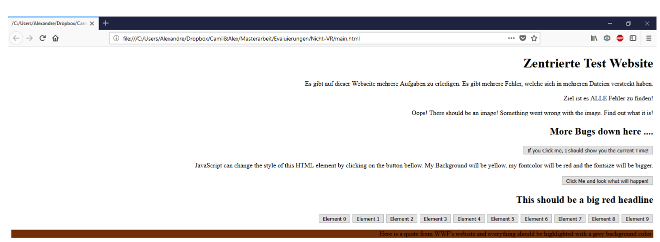


Figure 17. For non-VR: webpage with injected defects.

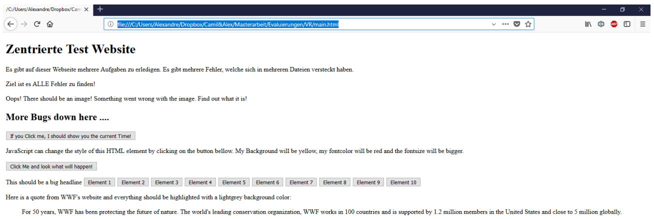


Figure 18. For VR: webpage with injected defects.

Figure 19. For non-VR: corrected webpage.

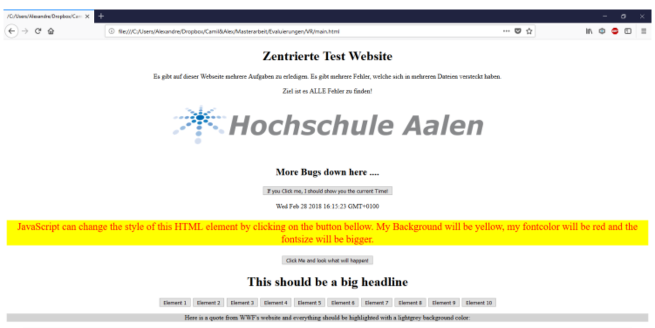


Figure 20. For VR: corrected webpage.

We asked them for a self-assessment of their HTML, CSS, and Java Script competency, the results of which are shown relative to their performance in Figure 21. To determine if the task order of VR or non-VR affected the results distinctly, each subject was randomly assigned to either begin with VR or with non-VR, and subjects 1, 5, and 8 initially began without VR. The total time needed until self-indicated completion was measured in seconds, and then this total duration was divided by the number of errors found (and corrected) to get an average duration per error for VR and non-VR separately. This is also shown in Figure 21. When comparing these average durations with the competency self-assessment, we see some correlation, with subject 1 who indicated little to no experience having some of the longest times, with subject 3 and 5 that have some experience have the second cluster of longer times, and subjects 2, 7, and 8 indicating the most competency and having some of the shortest times. Note that subject 1 was unfamiliar with VR, and unfamiliarity with this new environment may also have negatively affected this longest VR time; some prior training sessions may have reduced the duration. However, since the same subject is being compared in the two modes, the subject's programming and debugging experience and competency self-assessment should not significantly impact their own performance relative to themselves. Rather, the independent variable will likely have the greater effect.

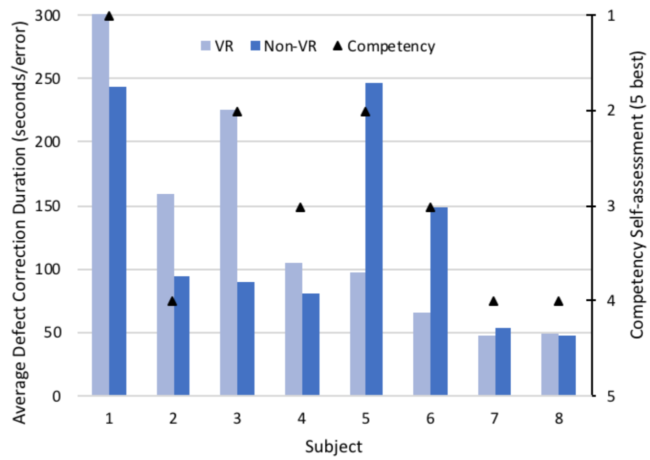


Figure 21. Competency self-assessment and average defect correction duration results for each subject in both the VR and non-VR setting.

As to task training or familiarity effects, as seen in Figure 21 the three subjects that began with non-VR tasks (1, 5, and 8) were slower on VR, faster on VR, and equivalently fast respectively. The other subjects also showed no bias trend with one or the other always faster. Thus, we conclude that there is little to no side effect as to task training in their ability to complete the task faster in the second environment.

TABLE I. HYDE-MR EFFICIENCY AND EFFECTIVENESS IMPROVEMENT VERSUS NON-VR

Subject	Experience[a]	HyDE-MR Improvement	
		Efficiency	*Effectiveness*
1	1	-23%	-13%
2	4	-69%	0%
3	2	-154%	0%
4	3	-30%	13%
5	2	60%	0%
6	3	56%	13%
7	4	11%	0%
8	4	-2%	0%
Average		-9%	13%

a. Scale of 1-5 (5 best)

Table I shows the performance difference of HyDE-MR relative to non-VR for the dependent variables efficiency and effectiveness. On average VR was overall 9% slower (97 seconds) and effectiveness was improved by 13%. However, these are only slight variations Conceivably this might could be attributed to the subjects having less total experience in

VR or to the margin of error given the relatively small population sample size. We also assume that because of the relatively short sessions of less than 30 minutes, that no significant performance impacts are attributed to task fatigue. Subjects 7 and 8 show almost no performance difference with HyDE, whereas for subjects 5 and 6 performance improved, and for 1-4 performance was mostly worse.

In the non-VR scenario, the subjects had access to one display but could use multiple windows. In the VR scenario, we observed them creating and using 4-5. After completing the tasks, subjects were debriefed as to how intuitive, suitable, and enjoyable HyDE was based on a Likert scale of 1 to 5, with 5 being very high and 1 being very low. The results are shown in Figure 22, where all the results were either positive or neutral (except one regarding intuitiveness). This indicates that the HyDE solution approach had a positive or neutral effect on subjective factors. None of the subjects reported VR sickness symptoms, a type of visually-induced motion sickness exhibiting disorientation [27], despite the inclusion of MR keyboard and mouse in VR and ongoing multiple display refresh. We also asked how many VR displays they would consider utilizing in they were given the opportunity, and their preference is shown in Figure 23, where we see that they could see themselves using up to 7 or 9 displays, whereas for non-VR the majority would use 3 real monitors if they could. This indicates that the subjects understood the potential of HyDE. When asked what environment they preferred, 75% preferred the VR over the classic non-VR environment (see Figure 24). Even those who preferred the classic environment liked the HyDE solution concept and prototype and think it has potential.

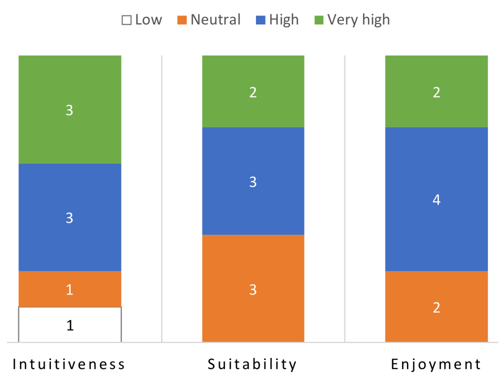


Figure 22. Rating given by the number of subjects for intuitiveness, suitability, and enjoyment of the HyDE prototype.

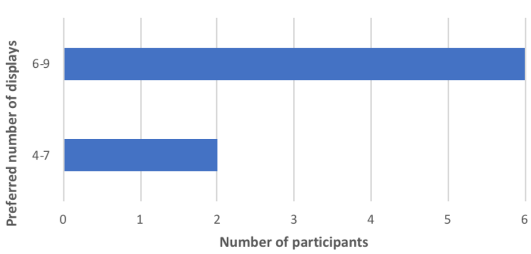


Figure 23. Preferred number of VR displays.

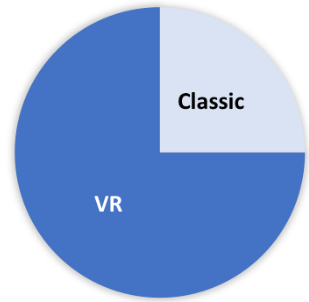


Figure 24. Subjects' preferred environment after the experiment.

As to interpreting the results, a convenience sample can obviously contain a number of biases, including under- or overrepresentation. The motivation of each individual at any point in time to find and fix a defect is another unknown factor, and we provided no reward system which gamification could induce. The differences observed in efficiency and effectiveness between HyDE on non-VR are in our opinion negligible, and we weight their subjective responses and the 75% preference of VR-based HyDE as an initial indicator of support for our solution approach or at a minimum an openness to utilizing VR for software development tasks. Further investigation is still needed.

VI. Conclusion

As VR devices become ubiquitous, it is only a matter of time before programmers wish to utilize VR capabilities for their development tasks as well. Our HyDE prototype demonstrated that the HyDE-VR and HyDE-MR hyper-display solution approach is feasible and can be a viable alternative to desktop displays. The HyDE-MR variant enables touch typing and the use of the mouse for screen interaction where appropriate, enabling programmers to interact more naturally for their code-centric programming tasks in the VR environment while remaining immersed. They thus can avoid interrupting their VR experience to take of the goggles and do programming changes and then put the VR gear on again. The HyDE concept is also generalized such that it can be applied to various domains beyond software development requiring simultaneous viewing and/or interaction of informational screens. The domain software development was selected to show HyDE's potential in intensive informational screen settings where software and tool preferences are non-uniform and support of and access to a large spectrum of software is imperative.

The evaluation based on a case-study and an empirical evaluation showed that the effectiveness in finding bugs was on par (1 bug more for one person), and although the sample size was small, the average task efficiency in VR was only slightly worse (-9% or 97 seconds per defect), which can be considered to be within the margin of error given the subjects' first use of this environment, their competency level for fixing these types of defects, and the sample size.

Future work includes a comprehensive empirical study including industrial usage, the inclusion of additional features, and performance optimizations.

References

[1] R. Oberhauser, “Immersive Coding: A Virtual and Mixed Reality Environment for Programmers,” In: Proceedings of The Twelfth International Conference on Software Engineering Advances (ICSEA 2017), IARIA XPS Press, 2017, pp. 250-255.

[2] C. Metz, *Google Is 2 Billion Lines of Code—And It’s All in One Place*. http://www.wired.com/2015/09/google-2-billion-lines-codeand-one-place/ [retrieved 2018.02.28]

[3] GitHub. https://github.com [retrieved 2018.02.28]

[4] G. Booch, "The complexity of programming models," Keynote talk at AOSD 2005, Chicago, IL, Mar. 14-18, 2005.

[5] PayScale, Full List of Most and Least Loyal Employees. http://www.payscale.com/data-packages/employee-loyalty/full-list [retrieved 2018.02.28]

[6] C. Jones, "The economics of software maintenance in the twenty first century," Version 3, 2006. [Online]. Available from: http://www.compaid.com/caiinternet/ezine/capersjones-maintenance.pdf 2017.02.23

[7] R. Minelli, A. Mocci, and M. Lanza, "I know what you did last summer: an investigation of how developers spend their time," Proc. IEEE 23rd International Conference on Program Comprehension, IEEE Press, 2015, pp. 25-35.

[8] L. Kappelman, "Some strategic Y2K blessings," Software, IEEE, 17(2), 2000, pp. 42-46.

[9] M.J. Pacione, M. Roper, and M. Wood, “A novel software visualisation model to support software comprehension,” Proc. 11th IEEE Working Conference on Reverse Engineering. IEEE, 2004, pp. 70-79.

[10] Dr. Dobb’s. Jolt Awards 2015: Coding Tools. http://www.drdobbs.com/joltawards/jolt-awards-2015-coding-tools/240169420 [retrieved 2018.02.28]

[11] RebelLabs Developer Productivity Report 2017

[12] S. Schuermans, M. Wilcox, L. Hecht, and C. Voskoglou. Developer Economics: State of the Developer Nation Q3 2017. SlashData, 2017.

[13] J. Steuer, “Defining virtual reality: Dimensions determining telepresence,” Journal of communication, 42(4), 1992, pp.73-93.

[14] D. Larimer and and Bowman, “VEWL: A Framework for Building a Windowing Interface in a Virtual Environment,” Proc. of Int. Conf. on Human-Computer Interaction Interact '2003. IOS Press, 2003, pp. 809-812.

[15] A. Febretti et al., “CAVE2: a hybrid reality environment for immersive simulation and information analysis,” The Engineering Reality of Virtual Reality, Vol. 8649 [864903]. International Society for Optics and Photonics, 2013.

[16] Z. Gu, D. Schleck, E.T. Barr, and Z. Su, “Capturing and exploiting IDE interactions,” Proceedings of the 2014 ACM International Symposium on New Ideas, New Paradigms, and Reflections on Programming & Software. ACM, 2014, pp. 83-94.

[17] A. Bragdon et al., “Code bubbles: rethinking the user interface paradigm of integrated development environments,” Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering-Volume 1. ACM, 2014, pp. 455-464.

[18] J. I. Maletic, J. Leigh, and A. Marcus, “Visualizing software in an immersive virtual reality environment,” 23rd Intl. Conf. on Softw. Eng. (ICSE 2001) Vol. 1, IEEE, 2001, pp. 12-13.

[19] F. Fittkau, A. Krause, and W. Hasselbring, "Exploring software cities in virtual reality," IEEE 3rd Working Conference on Software Visualization (VISSOFT), IEEE, 2015, pp. 130-134.

[20] Diehl, J. and Borchers, J., Tangible Windows. Technical report, RWTH Aachen University, 2013. URL http://hci. rwth-aachen. de/diehl.

[21] G.A. Lee, D. Nelles, M. Billinghurst, and G.J.Kim, "Immersive authoring of tangible augmented reality applications," Proc. of the 3rd IEEE/ACM international Symposium on Mixed and Augmented Reality, IEEE Computer Society, 2004, pp. 172-181.

[22] M. Billinghurst and H. Kato, "Collaborative mixed reality," Proc. First International Symposium on Mixed Reality (ISMR ’99), Springer Verlag, 1999, pp. 261-284.

[23] U. Neumann, S. You, J. Hu, B. Jiang, and J.W. Lee, "Augmented virtual environments (ave): Dynamic fusion of imagery and 3d models," IEEE Proc. Virtual Reality. IEEE, 2003, pp. 61-67.

[24] S. Prince, A.D. Cheok, F. Farbiz, T. Williamson, N. Johnson, M. Billinghurst, and H. Kato. "3d live: Real time captured content for mixed reality," Proc. International Symposium on Mixed and Augmented Reality (ISMAR 2002). IEEE, 2002, pp. 7-317.

[25] H. Kato and M. Billinghurst, "Marker tracking and hmd calibration for a video-based augmented reality conferencing system," Proc. 2nd IEEE and ACM International Workshop on Augmented Reality (IWAR'99). IEEE, 1999, pp. 85-94.

[26] Gupta, Shilpi, and Christopher Jaynes. "The universal media book: tracking and augmenting moving surfaces with projected information." In Proceedings of the 5th IEEE and ACM International Symposium on Mixed and Augmented Reality, pp. 177-180. IEEE Computer Society, 2006.

[27] R.S. Kennedy, K.M. Stanney, and W.P. Dunlap, “Duration and exposure to virtual environments: sickness curves during and across sessions,” Presence: Teleoperators & Virtual Environments, 9(5), 2000, pp. 463-472.

iaria journals

www.iariajournals.org

International Journal On Advances in Intelligent Systems
issn: 1942-2679

International Journal On Advances in Internet Technology
issn: 1942-2652

International Journal On Advances in Life Sciences
issn: 1942-2660

International Journal On Advances in Networks and Services
issn: 1942-2644

International Journal On Advances in Security
issn: 1942-2636

International Journal On Advances in Software
issn: 1942-2628

International Journal On Advances in Systems and Measurements
issn: 1942-261x

International Journal On Advances in Telecommunications
issn: 1942-2601