International Journal on Advances in Internet Technology

2013 vol. 6 nr. 3&4

The *International Journal on Advances in Internet Technology* is published by IARIA.
ISSN: 1942-2652
journals site: http://www.iariajournals.org
contact: petre@iaria.org

Sponsored by IARIA
www.iaria.org

International Journal on Advances in Internet Technology
Volume 6, Number 3 & 4, 2013

Joel Rodrigues, Instituto de Telecomunicações / University of Beira Interior, Portugal
Simon Pietro Romano, University of Napoli Federico II, Italy
Michele Ruta, Politecnico di Bari, Italy
Jorge Sá Silva, University of Coimbra, Portugal
Farzad Salim, Queensland University of Technology, Australia
Sébastien Salva, University of Auvergne, France
Ahmad Tajuddin Samsudin, Telekom Malaysia Research & Development, Malaysia
Josemaria Malgosa Sanahuja, Polytechnic University of Cartagena, Spain
Luis Enrique Sánchez Crespo, Sicaman Nuevas Tecnologías / University of Castilla-La Mancha, Spain
Paul Sant, University of Bedfordshire, UK
Brahmananda Sapkota, University of Twente, The Netherlands
Alberto Schaeffer-Filho, Lancaster University, UK
Peter Schartner, Klagenfurt University, System Security Group, Austria
Rainer Schmidt, Aalen University, Germany
Thomas C. Schmidt, HAW Hamburg, Germany
Didier Sebastien, University of Reunion Island, France
Zary Segall, Chair Professor, Royal Institute of Technology, Sweden
Dimitrios Serpanos, University of Patras and ISI/RC ATHENA, Greece
Jawwad A. Shamsi, FAST-National University of Computer and Emerging Sciences, Karachi, Pakistan
Michael Sheng, The University of Adelaide, Australia
Kazuhiko Shibuya, The Institute of Statistical Mathematics, Japan
Roman Y. Shtykh, Rakuten, Inc., Japan
Patrick Siarry, Université Paris 12 (LiSSi), France
Jose-Luis Sierra-Rodriguez, Complutense University of Madrid, Spain
Simone Silvestri, Sapienza University of Rome, Italy
Åsa Smedberg, Stockholm University, Sweden
Vasco N. G. J. Soares, Instituto de Telecomunicações / University of Beira Interior / Polytechnic Institute of Castelo Branco, Portugal
Radosveta Sokullu, Ege University, Turkey
José Soler, Technical University of Denmark, Denmark
Boyeon Song, National Institute for Mathematical Sciences, Korea
Victor J. Sosa-Sosa, CINVESTAV-Tamaulipas, Mexico
Dora Souliou, National Technical University of Athens, Greece
João Paulo Sousa, Instituto Politécnico de Bragança, Portugal
Kostas Stamos, Computer Technology Institute & Press "Diophantus" / Technological Educational Institute of Patras, Greece
Vladimir Stantchev, SRH University Berlin, Germany
Tim Strayer, Raytheon BBN Technologies, USA
Masashi Sugano, School of Knowledge and Information Systems, Osaka Prefecture University, Japan
Tae-Eung Sung, Korea Institute of Science and Technology Information (KISTI), Korea
Sayed Gholam Hassan Tabatabaei, Isfahan University of Technology, Iran
Yutaka Takahashi, Kyoto University, Japan
Yoshiaki Taniguchi, Osaka University, Japan
Nazif Cihan Tas, Siemens Corporation, Corporate Research and Technology, USA
Alessandro Testa, University of Naples "Federico II" / Institute of High Performance Computing and Networking (ICAR) of National Research Council (CNR), Italy

International Journal on Advances in Internet Technology
Volume 6, Numbers 3 & 4, 2013

CONTENTS

E-Learning and Self-Assessment for Hands-On Labs in Higher European Education

Fritz Laux
Fakultät Informatik
Reutlingen University
D-72762 Reutlingen, Germany
fritz.laux@reutlingen-university.de

Thomas Connolly
School of Computing
University of the West of Scotland
Paisley PA1 2BE, UK
thomas.connolly@uws.ac.uk

***Abstract*—"Learning by doing" in Higher Education in technical disciplines is mostly realized by hands-on labs. It challenges the exploratory aptitude and curiosity of a person. But, exploratory learning is hindered by technical situations that are not easy to establish and to verify. Technical skills are, however, mandatory for employees in this area. On the other side, theoretical concepts are often compromised by commercial products. The challenge is to contrast and reconcile theory with practice. Another challenge is to implement a self-assessment and grading scheme that keeps up with the scalability of e-learning courses. In addition, it should allow the use of different commercial products in the labs and still grade the assignment results automatically in a uniform way. In two European Union funded projects we designed, implemented, and evaluated a unique e-learning reference model, which realizes a modularized teaching concept that provides easily reproducible virtual hands-on labs. The novelty of the approach is to use software products of industrial relevance to compare with theory and to contrast different implementations. In a sample case study, we demonstrate the automated assessment for the creative database modeling and design task. Pilot applications in several European countries demonstrated that the participants gained highly sustainable competences that improved their attractiveness for employment.**

Keywords*—*learning by doing, virtual laboratory, hands-on lab, e-learning concept, self-assessment, e-grading.

I. Introduction

Effective knowledge transfer at Higher Education (HE) institutions and Vocational Educational Training (VET) should be tailored to the needs of its clients. VET, in contrary to HE, comprises of all non-academic professional education and training provided either by state or private organizations, e.g., in-company training. It is characterized by teaching practical skills that are needed in the daily work of employees, e.g., learning how to use a specific software product like a relational database product or how to normalize a relation.

Employees are highly motivated to acquire new skills but are often hindered to follow a scheduled training program. Students face a denser curriculum due to the Bologna process with a high degree of optional courses whose schedules and prerequisites are not aligned. Therefore, it is essential to provide self study courses with small module sizes to enable the participants to learn in their spare time at their own pace. In addition, in financially difficult times, knowledge transfer should be highly scalable in terms of costs. E-learning offers this capability but has the difficulty of keeping motivation high and to impart in-depth knowledge. Another challenge is the automated assessment of higher level understanding, a key requirement for large scale on-line courses. In engineering disciplines, e-learning has to deal with skills how to use commercial software or other technical devices.

As consequence, e-learning has to solve a multidimensional problem. Laux et al. [1] identified and presented at ICIW 2012 the following problem dimensions:

- content granularity
- theory level
- technology
- pedagogy
- assessment
- competence level

This leads to the following challenges. The learning content needs to be sliced into "digestible" portions while keeping the necessary context. Technological reality has to match and sometimes contrast with the theoretical underpinning. Technological aspects in Information and Communication Technology (ICT) are of particular importance to empower students and employees for a competitive labor market. Aiming for technological competence will stimulate the secondary motivation of the learners.

In our case study, which extends the results of [1], we focus on one of the most important areas in ICT competency for information management professionals: *database management systems* (DBMS). Databases are now the underlying framework of information systems that have fundamentally changed the way organizations and individuals structure and handle information.

Two crucial competences within the database domain are, how to structure efficiently a database and how to correctly process the data. For example, in the case of a banking application the database has to process the financial transactions correctly and reliably under any circumstances. This requires a sound understanding of the theory of transaction management and practical skills of software products at the same time. Such a highly specialized knowledge cannot be only theoretically taught neither could it be trained only by examples like a cookbook.

A second example is the database design, where profound understanding of the database model and the business domain is necessary in order to create a database structure that is efficient, resilient, and flexible for future change. It requires a clear view of the important entities of the business and its requirements, but also leaves room for abstraction and individual viewpoints. This makes the assessment of a data model difficult and subjective.

Both examples are scenarios for our e-learning concept with hands-on labs where we demonstrate how to teach effectively theory combined with hands-on labs for practical skills and problem solving competence.

A. *Structure of the Paper*

With the following overview on related work in cognitive science the context for our learning theory will be settled. In Section II, we point out the pedagogical requirements, the modularization constraints dictated by the learning objects and the tension between industry demands and long term knowledge for the students. This clarification is used in Section III as criterion for developing a unique reference model for the example learning object *database systems*.

With the help of Bloom's Taxonomy, we identify different knowledge levels in Section IV and show how to assess the higher levels of understanding.

Section V describes the supporting technology, in particular, the environment for the hands-on labs. We evaluate the e-learning reference model and discuss our findings during pilot runs of the learning modules in Section VI. The paper ends with a conclusion and ideas for future work.

B. *Related Work*

E-learning is a promising research subject and there is an abundance of publications on the foundation of on-line learning (e.g., [2][3][4][5][6]) as well as on its problems. For instance, the decreasing motivation was described by Prenzel [7] and Paechter et al. [8]. This is also confirmed by our own experience with e-learning.

According to constructivism [9] the learner generates knowledge by individual experience (radical constructivism [10]) or by social interaction within a cultural context (social constructivism [3]). As consequence, knowledge should be acquired by the learner in authentic situations that keep motivation high [11]. Connolly and Begg [12] report similar experiences and recommend teaching database analysis and design in a problem based environment.

Communication with fellow students and team work are also factors that support learning motivation [5]. This makes a communication and collaboration tool an indispensable ingredient of an e-learning system.

Multimedia support through e-learning systems is an enabler for flexible and scalable HE and VET, but is no guarantee for a successful on-line course. Critical voices raised the issue of superficial and routine knowledge that may easily be transferred. This knowledge refers to the cognitive domains one (remember), two (understand), and three (apply) of the revised Bloom's taxonomy [4].

In this taxonomy, Bloom [13] and Anderson [4] distinguish six cognitive levels

1) remember
2) understand
3) apply
4) analyze
5) evaluate
6) create.

For a deeper understanding the learner should acquire the higher cognitive levels. But, profound insights (analysis, synthesis/creation, and evaluation in Bloom's categories) are difficult to convey with a computer based learning environment as the study conducted by Spannagel [14] reveals. Krathwohl [15] gives a concise comparison of the original and revised taxonomy also stressing that the new taxonomy has four dimensions - factual, conceptual, procedural, and meta-cognitive knowledge.

The first three cognitive levels include the substance of subcategories of knowledge in the original framework. New is the meta-cognitive knowledge, which provides a distinction between knowledge and cognition in general as well as awareness of and knowledge about one's own cognition. This was not widely recognized at the time the original scheme was developed but is now of "increasing significance as researchers continue to demonstrate the importance of students being made aware of their meta-cognitive activity, and then using this knowledge to appropriately adapt the ways in which they think and operate" [15].

The Bloom/Anderson knowledge taxonomy was chosen because it structures knowledge according to the level of understanding and it fits well into the evaluation of skills-related learning. It is possible to distinguishes between the ongoing formative assessment (giving feedback about the student's performance or the assessment of educational materials during the course) and the summative assessment (evaluation at the end of the instructional cycle). So far, there are a couple of articles regarding e-learning assessment. Richards and DeVries [16] use the formative evaluation to dynamically monitor the learning activities in order to improve its course design. Their work focuses on the instructional design methodology and uses embedded questionnaires for the feedback. Velan, Jones, McNeil and Kumar [17] show in detail, how continuous online formative assessments helped medical science students to achieve better grades.

Experiences with summative assessment are reported by Chew, Jones and Blackey [18]. They introduced a range of of online assessment tools, such as electronic submission, partial tutor-intervention or a complete end-to-end computer-assisted assessment, at their university. As result from their experiences they recommend seven practices to follow and eleven to avoid. All these practices concentrate around organizational or technical aspects and how to gain a positive attitude towards e-learning and e-assessment. The real assessment work and feedback was based on closed questions that cover only the lower levels of understanding.

An approach for automatic marking of short essays from graduate students in computer science is described by Thomas, Haley, deRoeck and Petre [19]. They use a technique called

Latent Semantic Analysis (LSA) to infer meaning from a natural language text. This type of association analysis was used for marking free-form short essays. LSA produced - depending on the question - between 83% and 66% similar marks to an experienced human marker. For only one question out of six a significant statistical correlation was found.

A more promising result was achieved be Higgins and Bligh [20] when they applied computer based formative assessment in a diagram based domain. Compared to LSA this is understandable as the rules for diagram based models are far simpler and more precise by definition than a natural language. In their setting they assigned to a self programmed diagramming tool an explanation and feedback text for every diagramming rule. It was possible to add composite rules or features tailored for a specific task and context. From this approach raises a problem based on the fact, that each feature is assessed exactly once. It turned out that "several equally valid model solutions with slightly differing, mutually exclusive, features" [20] could not be assessed appropriately.

In our paper we will use state of the art products, like Oracle SQL Developer, MS SQL Management Studio, or Aqua Data Studio Entity Relationship Modeler. We do not want to insist on a preset model solution. The e-assessment process should allow to assess innovative, original and unexpected design solutions.

Problem based learning helps students to keep motivation high. But it seems difficult to ensure that theory and the necessary abstraction are drawn from an example. There are concepts that try to overcome these problems with the use of multimedia technology [6]. Blended learning, for example, tries to combine classroom learning with e-learning [2, chap. 10 and 29]. Classroom teaching can provide for theory and the e-learning session practice the knowledge in the form of exercises or experiments. We apply this technique for our virtual laboratory workshops described in Subsection III-C. This hybrid learning does not ensure sustainable and deep understanding, but, a well thought concept may help to convey deep insights as Astleitner and Wiesner [5] point out.

Our concept aims further: it contrasts and reconciles theory with the reality of commercial software products. This is important because software professionals and experts need the competence to verify the real behavior of a database system for instance and compare it with the theory. As a consequence real products are necessary as training tools and for assessment. No learning concept, so far, has tried to deal with the peculiarities of commercial graphical modeling products and provide an automated assessment of the resulting model.

II. Problem Description and Contribution

The goal is to provide a highly modularized e-learning environment for the specific theoretical and practical needs of HE and VET in the domain of ICT. For the proof of concept we have chosen the material produced during two EU funded projects: DBTech Pro (funded by the Leonardo da Vinci Programme) and its successor DBTech EXT (funded by the EU Lifelong Learning Programme).

The first project, DBTech Pro (http://myy.haaga-helia.fi/~dbms/dbtechnet/project2002-05_en.HTML), started with a survey to find out the needs of ICT-industry with regard to database technology competencies. We identified important knowledge areas of *database systems* and syllabi for course units. Based on these findings a framework and syllabus for the essential knowledge areas was developed, covering the related database standards, specifications, technology trends and understanding of the mainstream DBMS products. Course modules, including laboratory exercises, were developed and pilot courses executed in all five partner countries (Finland, Greece, Germany, Spain, and United Kingdom). The experiences from the pilot tests have been evaluated by students and experts.

The successor project DBTech EXT (http://myy.haaga-helia.fi/~dbms/dbtechnet/DBTechExtDescr.pdf) was formed by 7 universities, 3 VET institutes, and 1 industry representative. It extended the work from the previous project with a learning reference model with knowledge taxonomy and laboratory environment described in Section III. The model integrates instructional, active, and constructive learning concepts, which are applied as appropriate by the learning subject. Focus was on the in-depth knowledge with hands-on labs for learning by doing and verifying theories based on e-learning technology. Example topics have been database design, transaction processing, and data mining. Again, all courses and hands-on labs have been evaluated by the participants and teaching experts. More information about both projects may be found at http://www.dbtechnet.org.

From a pedagogical view, we identify the following requirements for a successful on-line course:

- self controlled learning
- authentic problem oriented learning
- most importantly, cooperative learning
- self assessment
- feedback and evaluation

Self controlled learning is important because of the above mentioned time constraints and with regard to different precognition of the learners. For a high motivation it is necessary to pose authentic real world problems and to let the learners solve them as a team [12]. This requires state-of-the-art software as used in industry.

Cooperative learning has two positive effects, one for the learner and one for the teacher, as follows. Communication among the students and working in groups keep motivation high and yield better learning results. It strengthen personal confidence of the learner through the positive feedback from the team. From the teacher's view the communication provides feedback on the effectiveness of the teaching and exercise material. The assessment of solutions for real-world problems is easier to justify and the acceptance from the students is more likely. In addition, communication among students reduces teacher intervention.

Employers demand key competences and skills that are predominately conveyed by cooperative learning:

- ability to solve real world tasks (problem solving)
- knowledge about state-of-the-art technology

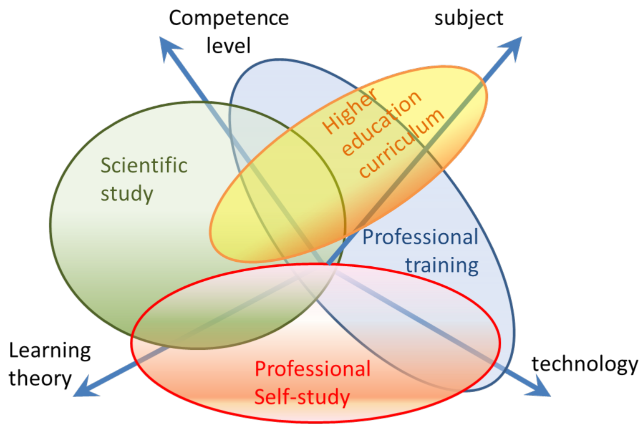


Fig. 1. Learning modules and its emphasis on target groups

- social skills, so called *soft skills*

Employees and students have increasing interest in learning skills that give a fast and easy to see return on their learning investment in form of directly applicable knowledge at their working place. This validates the first two requirements. Problem solving is a daily task at every workplace. Software problems most often can only be tackled successfully with state-of-the-art knowledge about technology. Social skills are indispensable for highly demanding ICT jobs [21] that involve cooperation and collaboration of many people.

In addition to the requirements mentioned above, the teaching units (modules), which we want to create, need to comply with the taxonomy of that domain, which defines how to decompose the content along the aspects:

- competence level
- subject area
- technology

Decomposing the content along the competence level provides different degrees of detail in line with target competencies and work profile. Students of HE institutions prefer a different learning concept than in VET courses. The latter have a tighter time schedule with less time for reflection of theoretical issues than HE students.

So, apart from the challenging content we tried to address all of the above requirements by dividing the learning content so that it can be combined and composed in multiple ways. Depending on the target learning group we produced different learning modules with specific consideration of the above described aspects. Figure 1 visualizes the emphasis of different target learning groups. For example, a module for professional training typically focuses on technology with a high level of competence and skills. Higher education curricula are less concerned about technology but concentrate on the subject theory. In contrast, professional self study needs to take special care of the didactic aspects as this is crucial for enabling self study and keeping the motivation high.

A. *Contribution*

The contribution of this paper consists of an integrated learning model for e-learning addressing the needs and constraints of HE and VET. For each learning unit the most appropriate learning theory was applied. Furthermore, the framework solves the problem of content modularization. Exemplary e-learning material that was used in multiple pilot runs proved the usefulness of this approach and resulted into more sustainable knowledge compared to traditional university teaching. The main advantage lies in the practical skills acquired using real DBMS products in the hands-on labs. The necessary lab environments are easy reproducible and provide full control of license restrictions. A major achievement of our approach is that we are able to assess automatically truly original and innovative data models produced by mainstream modeling tools. This is demonstrated with a commercial graphical Entity-Relationship diagramming and modeling tool.

III. The Reference Model

The reference model applies several learning concepts reflecting the different aspects and challenges presented in the previous section. The interrelation of these requirements make it difficult to optimize the learning concept. For better understanding we treat the dimensions content, lab environment, and project work separately and discuss the global optimization in Subsection III-E at the end of this section.

A. *Knowledge Taxonomy*

It is common to define a syllabus for the learning content. Structuring the syllabus results in a knowledge taxonomy of the teaching domain. From this structure we are able to deduce pre-requisites, identify learning elements, and designate learning outcomes. Structuring the teaching domain along the knowledge levels defined by Bloom [13] and Anderson [4] helped us to modularize the content according to knowledge depth and to provide teaching units for different target groups.

As an example, Figure 2 shows an extract of the DBTech database taxonomy [22] depicting the comprehension levels. Based on this layering we were able to deduce pre-requisites for every learning unit. For instance the unit *data modeling* (see Silberschatz et al. [23]) requires knowledge about the *relational, hierarchical*, and *network model*.

The knowledge levels are exemplified with the *transaction management*. On the lowest comprehension level it is sufficient to *recall* that transaction management coordinates transactions in a way that no undesired results may occur including the enduring protection of transaction results. For the second level the learner *understands* what this protection and coordination means in terms of the ACID[1] properties [24] and for level three the learner knows how to apply this knowledge. So far, it is not necessary to know how these properties are achieved by the transaction management. For the higher knowledge levels non functional aspects such as performance of different implementations of the ACID properties and its approximations are important. In order to *analyze* two concurrency mechanisms it is necessary to know their conceptual differences. Implementation knowledge is helpful to *evaluate* different mechanisms in terms of performance. For level six (*create*), the highest knowledge level of the Bloom/Anderson taxonomy, the learner needs a creative idea beside some experience in the implementation of a transaction management system.

[1]ACID stands for the characteristic properties of a transaction: Atomic, Consistent, Isolated, and Durable

	DBTech Pro Framework Reference courses and Topics	European Qualification Framework	IEEE/ACM CS2008	EUCIP initiative of CEPES	ACM AIS AITP IS 2002	BCS Professional Examination 2003	SweBOK
8							
9							
10	**Principles of Database Systems** (Level: Introduction, - obligatory)	Knowledge Level	CS2:IM1, IM2	EUCIP core IM knowledge		BCS Diploma (D) - Database Systems	CS 5 IM
11	Database Principle	Level 5	IM2.1, IM2.2	3.2.2.3		D 5.2	
12	Concepts of Database Systems and Environments	Level 3		3.2.2.1 3.2.2.4		D 5.1	
13	User roles	Level 3		3.2.2.4			
14	Ansi/Sparc Architecture	Level 3					
15	Conceptual models: ER and UML	Level 3					
16	Data Modeling	Level 5	IM2	3.2.2.2			
17	- Relational Model (RM) - Hierarchical (for XML) - Network Model (for ODBMS)					D 5.4 (RM)	
18	Relational Theory	Level 3					
19	- Relational Algebra	Level 2		3.2.2.6		D 5.4	
20	*Normalization*	Level 4				D 5.4	
21	Object-oriented Model	Level 5					
22	ODMG Standard	Level 5					
23	SQL Basics	Level 5		3.2.2.7			
24	QBE	Level 5					
25	Security	Level 3					
26	Transaction Processing	Level 4					
27	*Transaction Principle*	Level 4				D 5.6	

Fig. 2. Mapping of DBTech Database Taxonomy to other CS curricula (partial view) [22], sheet 2

B. Virtual Laboratory

The most important component of our e-learning model is the “learning by doing”. The concept of “learning by doing” became known in pedagogy through the work of Comenius [25]. From the perspective of developmental biology learning by doing is known even from animals [26] and experimenting (the systematic learning by doing) is fundamental in the development of the *homo sapiens* [27][28].

The psychomotoric learning keeps motivation high and supports a high degree of practical skills needed by companies. Moreover, the endurance of knowledge is much better and profound than without hands-on labs. Small, practical exercises and experimenting prepares the way for problem based learning.

In the case of ICT we have to deal with sophisticated, interdependent software systems like database management systems, application servers, data warehousing, On-Line Analytical Processing (OLAP) systems, or business intelligence suites. A student would need excessive time to install and set up the lab environment. This is unfeasible, considering only the risk that the system might be (unconsciously) misconfigured.

Another obstacle could be the different hard- and software equipment that might impede the installation of a certain product. But a realistic scenario consists of a suite of software products that must be configured in such a way that the programs can work together. For instance, an ERM-Modeling tool should produce a schema output that can be executed on the target database. The application server needs to cooperate with the database system and the web server. The only technical solution that actually works without problems is the virtualization technology. It provides a lab environment independent of the physical computer, which can be copied across the Internet to the learners’ computers. Even if a student accidentally damages the virtual system he can reset it to its original state. He is also able to save his results in a snapshot and continue later or at a different computer. There exist virtual image capturing and playing software that is freely available.

C. Virtual Laboratory Workshops

The technological complexity of the Virtual Laboratory makes it necessary to provide detailed, step-by-step tutorials for experimenting. The step-by-step tutorial is illustrated by screen-shots or from some complex procedures exist video sequences (animated screens). With this support the students can work through the lab experiments and design tasks without instructor intervention. Review questions allow the students to check their understanding and quizzes and open problems motivate for further investigations.

In order to make the learning more effective, we decided to use blended learning techniques and gather students for live workshops using the virtual laboratory. One trainer for about 10 students was sufficient to answer questions or to provide help with the virtual lab environment. Blended learning turned out to be effective for larger assignments or project work.

Between workshop sessions and for remote participants Skype telephone and remote assistance via web conferencing tools have been available. This allowed interactive help directly with the laboratory environment. The teacher could take over the students screen and demonstrate how to overcome a blocking situation.

The students had to submit their deliverables electronically via the e-learning platform for grading. The e-learning system was also heavily used as a discussion board, for self-assessment (see Section IV), and for feedback from students. The feedback was used to improve the presentation of the learning modules.

D. Project Work

While teaching theory in a didactic way and practicing or verifying the transferred knowledge in hands-on labs there is no guarantee that the students really acquire a problem solving competence. It is necessary to combine different knowledge pieces, then abstract and apply them as a whole. This systemic knowledge gap can be easily seen when students know about the ACID properties [24] of a transaction, but cannot relate a real world problem like the concurrent on-line reservation of flights with the concurrency issue. In the lab, such situations can be explored with real products and it is possible to test the behavior of the software in case of concurrent clients.

Moreover, students might be skilled in technological aspects of application servers but do not realize the danger of a compromised transaction due to technological tricks like pooled connections or disconnected components.

To ensure problem solving competences beyond technical issues students have to develop their ability to work in teams, manage tasks, organize releases and orchestrate different versions. All this knowledge can be learned from real world projects.

E. E-Learning Model

We believe it is best to decide from the learning content, which learning concept will be best suited for a specific content. The e-learning model we present integrates different learning concepts (see Issing [2]):

- Learning as behavioral modification for *practical skills* and verification of the theory
- Learning as active information processing using assimilation and accommodation processes to build a mental model of the *theory*
- Learning as construction of knowledge used for problem based learning as in *project work*

All these concepts are used in an integrative way in order to get the most effective results in terms of applicable knowledge and profound cognition that enable abstraction and problem solving to a large extent.

The design of the e-learning model (see Figure 3) starts with structuring the learning area guided by a taxonomy. The area is partitioned with a minimum of dependencies and each chunk of learning content is represented in a theory unit together with examples and demonstrations. Hands-on experiments help the students to verify and reflect the theory. At the same time they memorize situations and learn the necessary skills that help them to produce a solution for a related problem.

Examples and demonstrations explain the theory, making it easier to understand and familiarize with the concepts. Hands-on experiments motivate and stimulate students to reflect the theory. Examples provide the students with analogous situations that can be applied and abstracted in the project work.

The concrete real world problem forces the students to abstract from examples and construct a model of the problem

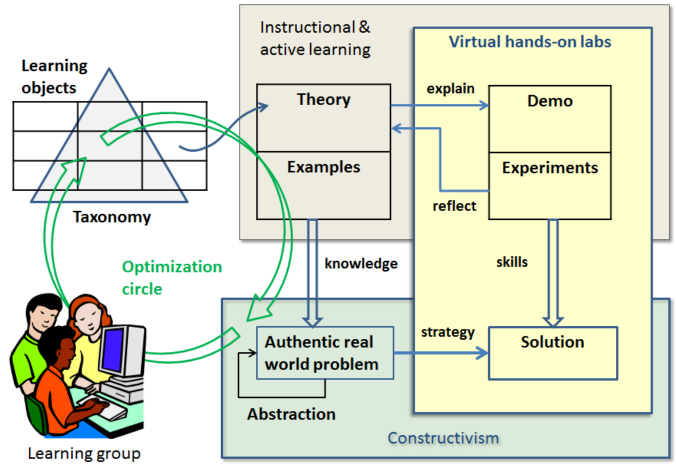


Fig. 3. E-Learning Model Overview

world in order to find a solution. Assessment of knowledge, skills, and solutions is driven by the expected competences that the learner should gain.

The interrelation of all these elements provided in a virtual lab environment with theory units, examples, and experiments are as shown in Figure 3. The global optimization task for the teacher is to select the learning objects, demos, examples, and experiments that lead to knowledge and skills necessary for solving the authentic real world problem and to put together all aspects in balance with the target learning group.

IV. ASSESSMENT

Knowledge levels 1 - 2 of Bloom's Taxonomy [13][4] seem easy to assess with multiple choice questions. Level 3 could be checked with a cloze test. The cloze text should refer to application knowledge that requires students to fill the gaps in the text with words such that the text gives meaningful instructions for a task. In case of programming language skills the assessment of level 3 knowledge could be automated by running the submitted source code against a compiler and executing the result. This is what we usually do when we assess and grade the students' knowledge of SQL. This procedure has the potential for self-assessment. The student enters the SQL-Query into a database system. Then, parser and syntax checker verify that the query is syntactically correct. If there is a syntax error, the student gets feedback in the form of a descriptive error message. When there is no syntax error the query is executed and the output can be compared with the expected result.

If the e-learning system supports a programming interface to the database system, then the assessment can be automated. This approach is used by SQL Training Programs like SQLZoo (http://sqlzoo.net/) by Andrew Cumming [29], SQL Tutor (http://www.cosc.canterbury.ac.nz/tanja.mitrovic/sql-tutor.html) by Antonija Mitrovic [30], the GNU SQLTutor (http://sqltutor.fsv.cvut.cz/cgi-bin/sqltutor), or SQL Trainer (http://www.inf-classic.fh-reutlingen.de/bisic) by Tomislav Bisic (Bachelor Thesis, Reutlingen University, 2008).

There are proven techniques how to grade the results taking into account guessing and "exclusion reasoning" [31]. For

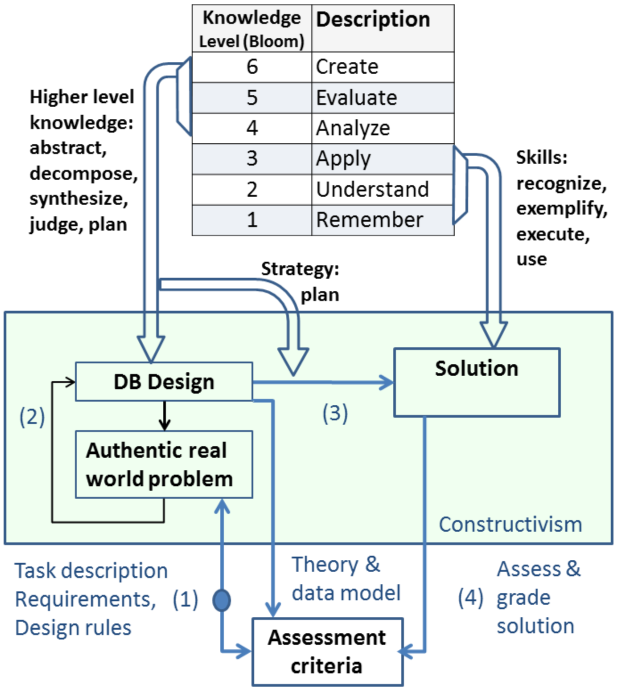


Fig. 4. Interrelation of knowledge levels to problem solving (constructivism) and assessment

instance the use of the correct aggregate functions in SQL should only count if the appropriate grouping is chosen for the query. For an unnecessary complex solution a penalty could be applied as those solutions tend to be pure guesses or the result of copying solutions from a similar task.

Much more challenging to assess are creative tasks and real project work. Here we have to assess artifacts that are the result of an analysis, or an evaluation, which flows into a creative process that eventually leads to a new solution. Especially design tasks like a data model, an application design, or a software architecture model are of this sort and cannot be measured only by the degree of fulfillment of the requirements. Further, important design criteria like simplicity, homogeneity, elegance, and structure of the solution are hard to put into a metric scale. Figure 4 shows the interrelation between knowledge levels and constructivist learning applied to problem solving with the focus on database design.

The approach by Kulkarni and Klemmer [32] includes detailed rubrics for the assessment. They break down the problem to a smaller granularity, but the main problem how to compare and assess different creative solutions in a reproducible way remains unsolved in our opinion. In their paper they provide rubrics and use a mix of students' self-assessment and assessment by the teacher. The example guidelines they give contain sentences like "The storyboards are hard to follow ..." or "The storyboards reasonably address the point of view ..." or "The storyboards are easy to follow ...". Our experience with these general type of rubric leads to a rather individual assessment result. What is "easy to follow" for one grader seems "hard to follow" for another. It mainly depends on the knowledge background of the grader and his ability to put himself into the position of the submitting student.

Kulm [31] recommends involving students in the construction of a scoring rubric. This not only helps to get a higher acceptance of the assessment but also makes the learning goals more transparent. Nevertheless, Ross et al. [33] argue that the students might not pay attention to the rubric if it is too general or to task-specific. If it is too general, it will fail to indicate what is essential in order to assess an artifact or a result. If it is to specific, it is too complicated for the students to use and it will hide the learning objectives or, it will lead to a non-adequate assessment of innovative solutions [33].

We have tried to reproduce the grading concept. But even when the rubrics where created together with the students, there was always a potential for disagreement whether a criteria was "reasonably addressed" or not and whether it "was easy to follow" or not. When we came up with criteria that were objectively measurable like: "Is it possible to query for a customer?" and "Does the query for a non-existent customer produce the message *customer not found*?". The number of criteria exploded and our impression was that a genuine innovative solution could not be foreseen and hence no appropriate question rubrics could be formulated in detail.

Fortunately, in the domain of database modeling the situation is not that hopeless because the data modeling task is underpinned and guided by a sound theory and a formal model. The data modeling task is given in form of a narrative description of the situation. The goal is to first produce a conceptual model, say an Entity-Relationship Model (ERM), and then transform the model to a normalized Relational Model (RM). Finally an SQL Schema will be produced for a specific database product that can be executed to create a database.

Figure 4 shows how different factors influence the assessment of database modeling.

The first step (1) from reading the task description to understanding what data will be necessary and need to be stored in the database is the most crucial. Sometimes, textbooks recommend identifying nouns and verbs which give hints what objects (nouns) to store and how they interrelate (expressed by the verbs). This works only partially as the description contains also activities and procedures that are applied to the data. The procedures are described by sentences using nouns and verbs, but neither of these are stored in the database, except possibly for their results. Therefore, some nouns and verbs have to be ignored, but others manifest itself as data in a database.

(2) The structure of the data depends on how the scenario is perceived. As an example take the address of a person. When we deal with that person as a customer, the address will be some property of the person. If the scenario is at the land registry office, the address of a lot is an entity of its own and the owner is only a property of the lot. In other cases it might be appropriate to have something in between like using the determining dependency between ZIP code and city.

(3) The outcome of the conceptual data model depends largely on the view of the task. This makes automatic assessment and grading of such a model difficult. Nevertheless, if two conceptual models with the same semantics are transformed to relational structures and normalized they will converge to the same normalized relational model. This proposition is supported by transformation rules, structural design patterns [23, chap. 7], and the relational normalization theory [34].

(4) Instead of manual assessing the conceptual data model (the ERM in our case), our idea was to rather assess and grade the transformed and normalized data model. In order to transform the model automatically to a relational model we used algorithms that applied heuristics, like transformation rules and design patterns. The normalization is done by a program that uses the semantics, i.e., functional dependencies, of the conceptual model. In the following subsection, we give some examples for how to transform different relationship types into relations.

A. One-to-many Relationship

One-to-many resp. many-to-one relationships are ubiquitous. This is also the reason, why there exists a number of synonyms for it, such as master-detail, body-feet, trunk-rootage, trunk-branches, one-level hierarchy. They may be used to model any kind of hierarchy: folder-files, house-rooms, order-items, bill of materials, collections, document structure, drainage system, etc. The structure could be expressed as ERM in various ways.

As an example, Figure 5 shows one-to-many (1:*) relationships in UML-notation that represent the same situation. All three model a customer order. Model (a) models the order in one single entity using user-defined data types (UDT) for the *customer* and *details* attributes. The *Details* attribute is a list of order details. Model (b) shows two entity types, *OrderHead* and *OrderDetail*. The *OrderHead* uses the a customer type UDT and *OrderDetail* are components of *OrderHead*. The third model (c) uses three entity types, one for the customer and two for the order. The relationships between Customer and OrderHead (*places*) as well as between OrderHead and OrderDetail (*contains*) are one-to-many. *OrderDetail* -a weak entity- depends on *OrderHead*.

If each model is transformed in a canonical way [23, chap. 7] into a relational model the result is similar to the ERM model (c) with foreign keys added to *OrderDetail* and *OrderHead* to represent the relationships *orders* and *contains*. In the case of ERM (a) the complex list data type of attribute *Details* is transformed to a separate relation connected by a 1:* relationship to the original *Order* relation. The UDTs used in model (a) and (b) need to be broken up into atomic data types in order to bring the relations into the first normal form. The result is shown as Bachman-diagram in Figure 6. If the model is further normalized to the third normal form (3NF) the relation *OrderDetail* will be broken up because of the functional dependency (FD) of *description* and *price* from *itemId*. This FD should be deduced from the scenario description.

B. Many-to-many Relationship

Another important structural pattern is the many-to-many relationship. The participation of employees in different projects is a good example for this pattern. An employee can work in many projects and a project will usually be worked on many employees. The worked hours shall be recorded per employee and project. This situation is modeled in the classical Chen-Notation in Figure 7. The entities can be directly transformed to relations. As the relational model can basically only represent 1:* relationships it is necessary to

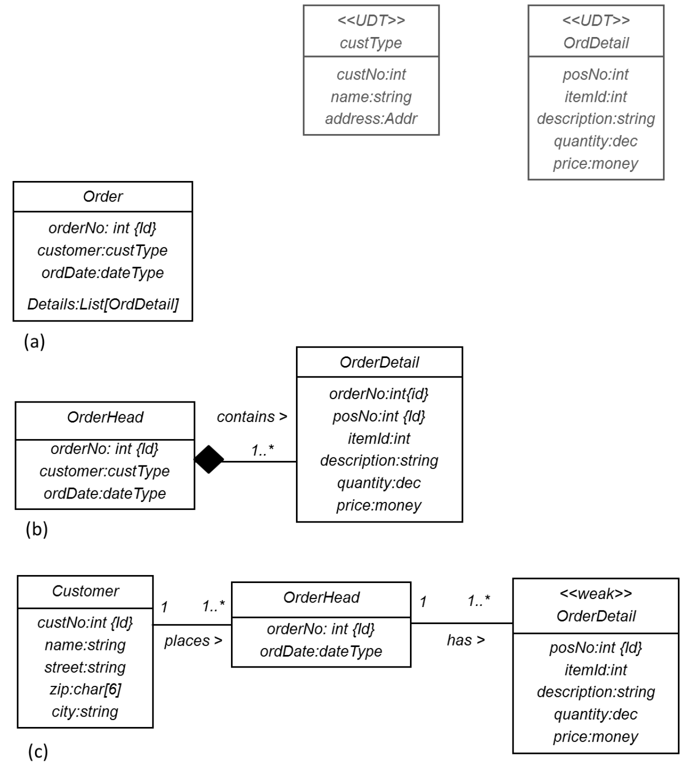


Fig. 5. Entity relationship models for a customer order. (a) one complex entity type using UDT (b) order composition (c) order split into 3 entities (Customer, OrderHead, OrderDetail)

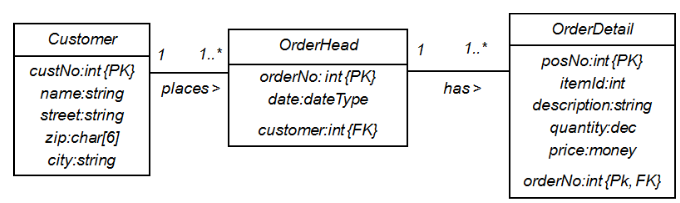


Fig. 6. Normalized relational model generated from the ER-Models shown in Figure 5

express the relationship in the form of a relation whose primary key is formed by the keys of *Employee* and *Project*. Both key attributes are foreign keys referring to *Employee* resp. *Project* as shown in Figure 8. The relationship attribute *workedHours* is added to the relationship table.

C. General Transformation Rules

The previous examples may be generalized in the following way:

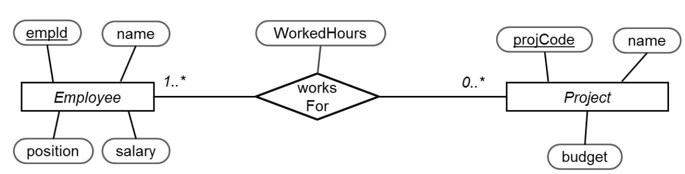


Fig. 7. Entity relationship model (Chen notation) with many-to-many relationship representing employees that work in different projects

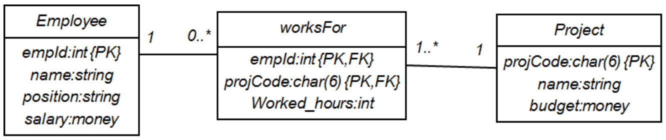


Fig. 8. Normalized relational model generated from the ER-Model shown in Figure 7

1) Map each entity type without attribute change to a relation type.
2) Transform each one-to-many relationship into a foreign key on the many-side. Special case one-to-one can be achieved by a foreign key that is a primary key as well. If the relationship has attributes then place them on the relation with the foreign key.
3) Construct a new relation for every many-to-many relationship using the primary keys from the connected entities as a compound primary key. Add the relationship attributes to the relationship table.
4) Construct a separate relation type for every collection type attribute and add an foreign key to this new relation pointing to its original relation.
5) Break up each complex UDT into its atomic data types in order to bring the relations into first normal form.
6) Normalize the relations into third normal form.

Applying these rules will transform equivalent entity relationship models into the same relational model. Even different viewpoints will merge to the same relational schema because of the normalization process. The final result of the given examples from Figure 5 is the normalized SQL schema listed in Figure 9. The aim of this SQL schema is that the graphical model is now available as textual schema that can be further assessed with the help text analysis methods.

The transformation process can be largely automated with data modeling tools provided by database vendors. To cope with possible naming variance it is possible to provide a list of relevant synonyms and its abbreviations. As example, we demonstrate the process by using the Oracle SQL Developer in the next subsection.

D. Practical Issues for Automatic Mapping an ERM to a normalized RM

Our goal is to automate the assessment of creative and analytic knowledge like the design of an entity relationship model. We want to show that the assessment of an ERM can be considerably automated by the help of standard database tools. Oracle's SQL Developer is a graphical entity relationship modeling tool. Figure 10) shows its graphical user interface. The graphical representation of the ERM is entered on the right pane. The elements of the ERM (entities, attributes, domains) are recognized by the tool and listed in the lower left pane of the display. The purpose for using this tool is threefold:

1) make students acquainted with a leading commercial modeling tool.
2) provide the students with a widely used rendering (crow foot notation) of the ERM.
3) allow the transformation of the ERM into a textual version (SQL schema).

```
CREATE TABLE Customer
  (
  custNo INTEGER NOT NULL primary key,
  name VARCHAR(22),
  street VARCHAR(22),
  ZIP CHAR(6),
  city VARCAHR(22)
  )
;

CREATE TABLE OrderHead
  (
   orderNo INTEGER  NOT NULL PRIMARY KEY,
   customer INTEGER NOT NULL,
   ordDate DATE,
   FOREIGN KEY customer REFERENCES Customer
             on delete restrict
  )
;

CREATE TABLE OrderDetail
  (
   posNo INTEGER  NOT NULL ,
   itemId INTEGER NOT NULL ,
   quantity NUMBER ,
   orderNo INTEGER  NOT NULL,
   FOREIGN KEY orderNo REFERENCES OrderHead
             on delete cascade,
   FOREIGN KEY ItemId REFERENCES Item
             on delete restrict
  )
;

ALTER TABLE OrderDetail
  ADD CONSTRAINT "OrderDetail PK"
  PRIMARY KEY ( orderNo, posNo ) ;

CREATE TABLE Item
  (
   itemId INTEGER NOT NULL PRIMARY KEY,
   description VARCHAR2 (44) ,
   price NUMBER(8,2)
  )
;
```

Fig. 9. SQL Schema in 3NF generated from the relational model of Figure 6

The student uses this tool to edit his ERM model of a given scenario. This model can be transformed to an SQL schema by just pressing a button. Structured data types are mapped to user defined data types (UDT). Multivalued attributes are mapped to an array type (SQL:1999). The schema should be normalized in order to match a normalized reference model. The normalization is done by a program that implements the normalization algorithm given by Vinek et al. [35]. Functional and multivalued dependencies are derived from a description of the modeling task. The dependencies are input to the program in the following form

<attrIds> -> <functional-dependent-attrIDs>.
<attrIds> ->> <multivalue-dependent-attrIds>.

The program handles multivalued dependencies and structured attributes. For assessment purposes, we count the number of matching elements. Primary and foreign keys are specially weighted to reflect their importance. We deal with possible naming variants by supplying an editable list of names for all relevant elements (entities, attributes, and relationships) and use their identifier (Id) instead. This is feasible, since a typical modeling assignment contains usually less than 30 names with at most 4 equivalent variants each. The synonym list for customer order is similar to Table I. With such a list we

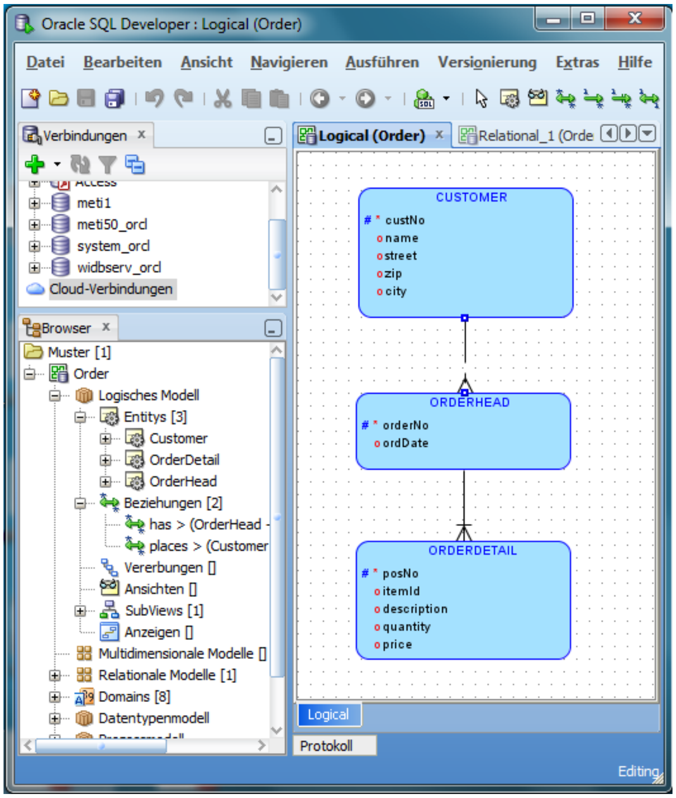


Fig. 10. Screen-shot of Oracle's SQL Developer showing the ERM of Figure 5 (c)

TABLE I. SAMPLE SYNONYM LIST FOR CUSTOMER ORDERS

Id	name1	name2	name3	name4
1	customerNo	customerId	custNo	custId
2	custName	name	cName	
3	ZIP	postCode	PLZ	
...	...	...	...	...
12	itemId	itemNo	productId	prodId
13	description	descr	itemName	
14	price	unitPrice	salesPrice	

identify and match automatically most synonyms that appear in students solutions.

V. TECHNICAL FRAMEWORK AND INFRASTRUCTURE

Our framework of technologies provides a central, web based repository for teaching material, lab environments, multimedia, communication and collaboration tools.

A. E-Learning Portal

We provide all e-learning material through a portal (see http://dbtech.uom.gr and [36]) using Moodle as the software platform. It contains all theory units, mostly as reading material, video lectures, tests, assessments and experimental lab environments that will be described in the following subsection. Local versions, like translations or modifications that fit the curriculum constraints, are hosted and maintained at the project partners' sites (https://relax.reutlingen-university.de for Reutlingen University, or https://elearn.haaga-helia.fi/moodle/login/index.php for Haaga-Helia University for Applied Sciences).

B. Virtual Laboratory Infrastructure

The lab environments are available either through technologies like desktop virtualization or virtual machines running computer software images. The latter is used when the image only uses free software. In this case, there is no need to control the number of downloads or to provide licenses. After downloading the image it can run off-line. Free players for the image are widely available, e.g., VirtualBox.

The virtual infrastructure contains first of all, a database management system (DBMS), like PostgreSQL or free versions of commercial systems like Oracle XE or DB2 UDB Express Edition. For the database modeling we use the free Oracle SQL Developer in conjunction with a normalization software that was developed by a student.

For commercial software products, which require licenses, the use of a desktop virtualization is more appropriate since it allows easy control of the number of remote application accesses. Citrix XenDesktop or VMware View are examples that provide a Virtual Desktop Infrastructure (VDI) for different operating systems.

VDI provides remote access to a pool of virtual machines through a connection broker. If the license policy is only for a certain number of concurrent users it is no problem to limit the concurrent users with this software. Access control may be enforced by LDAP or Active Directory. The virtual machines are automatically managed for every user in terms of multiple, customized instances of computer systems and applications. Independent virtual machines may be assigned to avoid any resource access conflicts. Access to different operating systems is possible and the assignment to a client's PC may be persistent or transient.

The virtual machines are accessed from a client machine via local or public area network. Client computers only need a web browser with ActiveX or Java Applet technology support. Such a support is given by the most common web browsers.

DBTech EXT uses a VDI operated by the University of Málaga. The number of concurrently active virtual machines depends on the resources (processor cores, memory, and disc space) provided. For the DBTech EXT labs Málaga uses two VMware servers with two quad-core processors and 32 Gigabytes of RAM each [37]. This infrastructure has enough power to run 96 concurrent virtual systems, each with 512 Megabytes of memory. The VDI architecture is presented in Figure 11 showing the VMware architecture consisting of a virtual center and two Hypervisor ESX servers that provide for multiple operating systems running on a single physical server. The broker is responsible for dispatching the connection requests from clients and to control the access with the help of an authentication service.

VI. EVALUATION AND EXPERIENCES

The experiences mainly stem from two EU funded projects described earlier that were carried out during the years 2002-2005 and 2009-2010 (see http://www.dbtechnet.org).

A couple of example e-learning modules have been developed as testing materials and these courses were used for virtual workshops conducted during and after the second project.

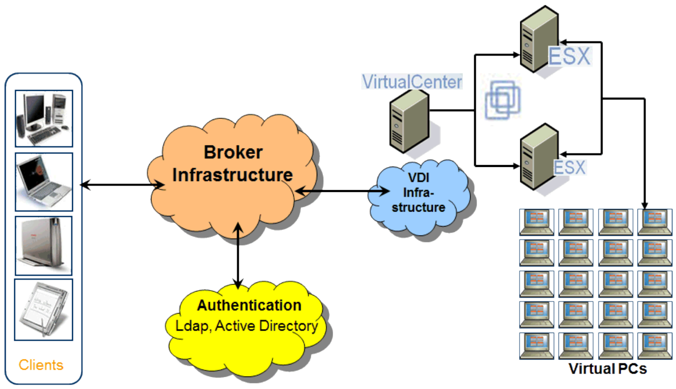


Fig. 11. Virtual Desktop Infrastructure for virtual labs [37]

For the virtual workshop the e-learning platform was enhanced by communication and collaboration tools like Skype, discussion boards and upload areas for deliverables. Teaching materials were structured and furnished with exercises and assignments for the students. The exercises used the previously described virtual infrastructure to guarantee a predefined and fully functional environment. Assessment of the students was done by on-line tests preferably in form of multiple choice questions.

In addition, self-assessment was provided for design tasks using a commercial data modeling tool in conjunction with a normalization software. The software supported semi-automatic grading and self-assessment for data models described in Section IV, which relieved the teacher from time consuming manual assessment.

The effectiveness of our hands-on labs was assessed by weekly online-tests. For evaluation purpose we divided the students into two groups: one group of 23 students took the traditional paper exercises and the other group of 17 students enrolled in the e-learning hands-on labs using the virtual lab environment.

Both groups participated at weekly self-assessment tests. The answers to the multiple choice questions were collected and assessed with the help of the e-learning system. However, the e-learning system provided only support for multiple choice questions to test the analytic skills and not the construction of knowledge or innovative solutions. As a consequence, additional tools were used in the virtual lab environment to assess creative tasks like data modeling.

It turned out, that the e-learning group with hands-on labs performed 28% better than the control group. The difference was even more apparent, when we ranked all students and found that 64% of the e-learning group were found better than the median, whereas only 41% of the control group ranked better than the median.

At the end of every semester the students have to pass written examination under supervision. We compared the results of our e-learning course students ($n = 17$) with previous year's students (our control group, $n = 36$) who were taught in the traditional way. The learning topics of the written examination were weighted according to its importance for the learning goal. For an easy comparison of the results we scaled the grades for each topic from 0 (least) to 100 (best). For example, a grade of 66 indicates that $2/3$ has been achieved.

The results from the weekly tests could not only be confirmed, but surpassed. The significance of the results was tested with Student's *T-Test* under the hypothesis "E-learning with hands-on lab is *not* superior to traditional learning" (H_0) and the standard deviation was tested with Fisher's *F-Test* with the hypothesis "The variance of the mean value of the results are the same" (H_0).

TABLE II. STATISTICAL EVALUATION RESULTS

criteria	e-learn $n = 17$	control $n = 36$	% diff of control	T/F test for $\bar{x}/\sigma$, $\alpha =$
$\bar{x}$ total grade	66	48	+37%	0.009
σ total grade	17.7	18.6	−4.8%	0.966
$\bar{x}$ design grade	76	48	+58%	0.0004
σ design grade	12.9	22.8	−52%	0.077
$\bar{x}$ SQL grade	74	46	+61%	0.0002
σ SQL grade	24	18	+33%	0.196
$\bar{x}$ TaMgmt grade	65	38	+37%	0.009
σ TaMgmt grade	34	26	+31%	0.187

The e-learning group achieved on average 37% better total grades than the control group (see Table II). The hypothesis H_0 was rejected and H_1 "E-Learning with hands-on lab is superior to traditional learning" was accepted. The standard deviation of the e-learning group was slightly smaller than for the control group, which indicates that the spreading of the learning success was similar.

Looking at the examination results in detail revealed that the e-learning group achieved 58% better grades for database design tasks and even better results (+61%) for their SQL competence. For the database design task H_0 was rejected with an error $\alpha = 0.0004$, which means that H_1 "E-Learning with hands-on lab is superior to traditional learning" was accepted with a confidence level of 99.96%. The standard deviation of the e-learning group was 52% smaller compared to the control group leading to the rejection of H_0 with an error $\alpha = 0,077$ and acceptance of H_1 "The variance of mean value of the results are different" with an confidence level of 92.3%. This leads to the statement that the learning success is more uniform when using hands-on labs.

The results for the SQL competence of the students with hands-on labs was even more impressing with 61% improvement (error $\alpha = 0.0002$, 99.98% confidence). The grades for the transaction management (TaMgmt) topic improved by 39%. In both cases, the learning improvement was accompanied by a larger standard deviation (+33% and +31%), which is in contrast to the result of database design topic. The result is not very significant ($\alpha = 0.196$ and $\alpha = 0.187$) but the reason for this result is unknown and needs further investigation.

The methodologies used to evaluate and assess our concept was not only statistical, but included informal and formal (survey conducted via the e-learning platform) feedback, and discussions with students and experts (professors and industrial trainers). The students pointed out the motivational aspect of real products. They claim higher competence and labor market attractiveness. A negative point was that the products are "black boxes" that allow no inside view. Students felt happy with the e-learning support material as this allowed them to check most of their solutions on-line.

The teachers/instructors indicated that some students took wrong reasoning from the products' behavior that was difficult to correct later. This was especially problematic, when a strong theoretical foundation was essential, as with normalization theory and serializability. Teachers appreciate that they had more time to discuss difficult questions with students. The time saved was attributed to the self-assessment capability of the virtual lab environment.

At Reutlingen University, study projects of real world problems have been incorporated into the curriculum for more than 10 years. Over many generations of students the feedback was uniformly positive. Students appreciate the real life character of the projects. In about one third of the projects, the problem was posed by a company that also collaborated with the student teams. From a didactic point of view the motivation was kept high if the company or the university committed itself to use the project results. In most cases, this was a software developed by the students.

Problem based learning confirmed the proposed high motivation, if the knowledge background of the project team was sufficient to master the problem. It was not necessary for every participant to have management competence or to be an expert programmer. It was sufficient to have at least one with the necessary capability. In most cases, this stimulated the team and resulted in an intensive team internal learning process. The supervising professor has the responsibility to make sure that the students with less knowledge will not get frustrated. A possible intervention could be an additional training for the "weaker" students or to assign a different role to the "dominant" student. In individual situations we have successfully been able to let more advanced students act as trainers for the group over a certain time period.

Comparing student teams that work physically together outperform teams that only work together virtually. In feedback discussions the students state a lower motivation and commitment to the project team if they worked remotely without meeting each other. Asking for reasons the students named the missing personal contact and commitment. In contrast, the teams that met regularly developed a culture of responsibility that supported motivation and contributed to the project success.

VII. Conclusion and Future Work

The outstanding lessons learned of this long term e-learning experience can be summarized in five statements:

1) A key success factor is the adequate decomposition of the knowledge domain. Only if this requirement is granted, the necessary small chunks of information are identified and can be prepared according to our e-learning model. If the chunks are not small and sufficiently independent it is hard to provide e-learning modules that can be worked through without the constant help of the teacher.
2) E-learning is not superior to face-to-face teaching but together with the hands-on lab it provides about 30% to 40% better learning results. The preparation of study material is much more elaborate than for traditional teaching.
3) E-assessment scales better only for lower comprehension level (Bloom's taxonomy) and partially for the application level.
4) For higher level (deeper understanding) as synthesis (create), evaluation and analysis, specific tools are necessary that are able to capture and measure the work results. For the case of *data modeling* we demonstrated this successfully.
5) Conducting real world projects with small groups of students provides the highest motivation and yields long lasting competence and enduring confidence for the students.

The e-learning method with hands-on lab should be generally applicable if the subject has a strong formal foundation. This is the case for engineering and formal sciences. For these domains it should be possible to create programs or machines which students can use and experiment with. Possible examples are computer aided electronic design, process simulation, and structural design.

Another challenge is the reuse of e-learning material. Even though there exists an e-learning standard, modules are usually technologically incompatible, e.g., a Moodle (https://moodle.org/ course unit cannot be used in a Blackboard (http://www.blackboard.com/Sites/International/EMEA/index.html) environment. The reuse of e-learning material might be further restricted by different versions of the same product or browser.

The question of self-assessing the learning outcome depends on the rubric. If the topic is formally grounded, sufficient criteria for a quantitative rubric should exist. If a satisfactory metric for the assessment can be found, it is only a question of implementation. Expert systems with explanation component can be used to assess the students artifacts and explain the grading or comment the students work.

The use of hands-on labs is applicable for engineering, ICT, and partly in natural sciences, where a kind of sandbox or virtual lab environment can be provided with design and simulation tools used in industry. Possible examples are architecture, mechanical engineering, electronics, and chemistry. In all these disciplines, the results of creative processes are depicted in a formal way (construction drawings, formal language, model, schemata, etc.), which can be analyzed and assessed automatically according to precisely defined criteria. Case studies for these disciplines are subject of further research.

It is less likely that creative tasks with low formal underpinning can be dealt in a similar way. The result of a complex project that involves different technologies and if the solution includes manual tasks will be still too hard to assess automatically. Examples for this type of projects are process optimization or product development. The general task to assess any kind of creative oeuvres still remains a challenge for future work.

References

[1] Fritz Laux, Martti Laiho, and Thomas Connolly, "E-Learning with Hands-On Labs in Higher Education", In F. Laux and P. Lorenz (eds.), *The Seventh International Conference on Internet and Web Applications and Services* (ICIW), pp. 231-237, Stuttgart, Germany, 2012; ISBN: 978-1-61208-200-4.

[2] Ludwig Issing, “Psychologische Grundlagen des Online-Lernens” [engl. Psychological Foundations of Online Learning] , in: Paul Klimsa and Ludwig Issing (eds.), *Online-Lernen - Handbuch für Wissenschaft und Praxis*, 2^{nd} ed., Oldenbourg V., München, 2011.

[3] André Kukla, *Social Constructivism and the Philosophy of Science*, London: Routledge, 2000; ISBN 0415234190.

[4] Lorin W. Anderson and David R. Krathwohl, (Eds.)(2001). *A Taxonomy for Learning, Teaching, and Assessing: A Revision of Blooms Taxonomy of Educational Objectives*, New York, Longmans, 2001.

[5] Hermann Astleitner, Iwan Pasuchin, and Christian Wiesner, “Multimedia und Motivation - Modelle der Motivationspsychologie als Grundlage für die didaktische Mediengestaltung” [engl. Multimedia and Motivation - Models of Motivation Psychology as Foundation for Didactic Media Design], Medienpädagogik, Zeitschrift für Theorie und Praxis der Medienbildung, 22.3.2006. [Online]. Available: http://www.rz.uni-frankfurt.de/internet/elearning_2008/pdf/Artikel_Astleitner_und_co.pdf, last access: 16.08.2013.

[6] Hermann Astleitner and Christian Wiesner, “An Integrated Model of Multimedia Learning and Motivation”. Journal of Educational Multimedia and Hypermedia, 13(1), pp. 3-21, 2004; ISSN 1055-8896.

[7] Manfred Prenzel, “Sechs Möglichkeiten, Lernende zu demotivieren” [engl. Six Possibilities to Demotivate Learners], in: H. Gruber and A. Renkel (eds.), *Wege zum Können, Determinanten des Kompetenzerwerbs*, Bern, Huber, pp. 32-44, 1997.

[8] Manuela Paechter, Karin Schweizer, and Bernd Weidenmann, “Lernen in virtuellen Seminaren: Neuigkeitsbonus oder Adaption an ungewohnte Lernbedingungen” [engl. Learning in Virtual Seminars: Novelty Bonus or Adaption to Unusual Learning Conditions], in: F. Scheuermann (ed.), *Campus 2000 - Lernen in neuen Organisationsformen*, Münster, Waxmann, pp. 279-288, 2000.

[9] Seymour Papert, *Constructionism: A New Opportunity for Elementary Science Education* , Massachusetts Institute of Technology, Media Laboratory, Epistemology and Learning Group, 1986.

[10] Ernst von Glasersfeld, “Konstruktion der Wirklichkeit und der Begriff der Objektivität” [engl. Construction of Reality and the Concept of Objectivity]; in: Heinz von Foerster: *Einführung in den Konstruktivismus*, publ. of Carl-Friedrich-von-Siemens-Stiftung, vol. 5; München: Piper, 1992; ISBN 3-492-11165-3.

[11] Jochen Gerstenmaier and Heinz Mandl, “Wissenserwerb unter konstruktivistischer Perspektive” [engl. Knowledge Acquisition under Constructivist Perspective], Zeitschrift für Pädagogik, 41. Jg, Nr. 6, pp. 867-888, 1995.

[12] Thomas Connolly and Carolyn Begg, “A Constructivist-Based Approach to Teaching Database Analysis and Design”, Journal of Information Systems Education (JISE), Vol. 17(1), pp. 43-53, 2006.

[13] Benjamin Bloom, Max Engelhart, Walker Hill, and David Krathwohl, *Taxonomy of educational objectives: the classification of educational goals*; Handbook I: Cognitive Domain, New York, Longmans, Green, 1956.

[14] Christian Spannagel, *Benutzungsprozesse beim Lernen und Lehren mit Computern* [eng. Processes for Learning and Teaching with Computers], PhD-Thesis, Hildesheim, Franzbecker, 2007; ISBN-10: 3-88120-443-9.

[15] David R. Krathwohl, “A Revision of Bloom’s Taxonomy: An Overview”, Theory into Practice, Volume 41, Number 4, Autumn 2002, College of Education, The Ohio State University.

[16] Griff Richards, Irwin DeVries, “Revisiting Formative Evaluation: Dynamic Monitoring for the Improvement of Learning Activity Design and Delivery”, In Proceedings of the 1st International Conference on Learning Analytics and Knowledge (LAK 2011), Banff, AB, Canada, February 27 - March 01, 2011.

[17] Gary M Velan, Philip Jones, H Patrick McNeil, Rakesh K Kumar, ”Integrated online formative assessments in the biomedical sciences for medical students: benefits for learning”, in BMC Medical Education, Volume 8:52, 2008; DOI: 10.1186/1472-6920-8-52.

[18] Esyin Chew, Norah Jones, Haydn Blackey, “Implementing Institutional Online Assessment Addressing the Challenges”, in Proceedings of: Hybrid Learning, Third International Conference (ICHL 2010), Beijing, China, August 16-18, 2010.

[19] Pete Thomas, Debra Haley, Anne deRoeck, Marian Petre, “E-Assessment using Latent Semantic Analysis in the Computer Science Domain: A Pilot Study”, Proceedings of the Workshop on eLearning for Computational Linguistics and Computational Linguistics for eLearning (eLearn ’04), pp. 38-44, 2004.

[20] Colin A. Higgins and Brett Bligh, “Formative Computer Based Assessment in Diagram Based Domains”, Proceedings of the 11th annual SIGCSE conference on Innovation and technology in computer science education, pp. 98-102, Bologna, Italy, 2006

[21] Luis Fernández Sanz, “Analysis of non technical skills for ICT profiles”, 5^{th} Iberian Conference on Information Systems and Technologies (CISTI), Santiago de Compostela, 2010.

[22] Dimitris Dervos, Martti Laiho, and Fritz Laux, “DBTech Pro Framework Reference Courses and Topics”, [Online] http://dbtech.uom.gr/mod/resource/view.php?id=592 (Excel-Format), last access: 16.03.2012.

[23] Abraham Silberschatz, Henry Korth, and S. Sudarshan, *Database System Concepts*, New York, McGraw-Hill, 2011.

[24] Gerhard Weikum and Gottfried Vossen, *Transactional Information Systems*, Morgan Kaufmann Publishers, 2002.

[25] Johann Amos Comenius, *Didactica Magna*, 1657, In: Klaus Schaller (ed.), Ausgewählte Werke. vol. 1, Hildesheim: Olms, 1973, p. 82.

[26] Gordon M. Burghardt, “The evolutionary origins of play revisited: Lessons from turtles”. In M. Beckoff and J. A. Byers (eds.), *Animal Play: Evolutionary, Comparative, and Ecological Perspectives*, Cambridge, UK: Cambridge Univ. Press, 1998.

[27] Doris Bergen (ed.), *Play as a Medium for Learning and Development*, Portsmouth, NH: Heinemann, 1987.

[28] Andreas Flitner, *Spielen - Lernen : Praxis und Deutung des Kinderspiels* [engl. Play - Learning : Practice and Interpretation of Children’s Game] , 3^{rd} ed., 1998.

[29] Dr Gordon Russell and Andrew Cumming, “Improving the Student Learning Experience for SQL Using Automatic Marking”, IADIS International Conference Cognition and Exploratory Learning in Digital Age (CELDA), Lisbon, Portugal, 2004, pp. 281-288.

[30] Antonija Mitrovic, “An Intelligent SQL Tutor on the Web”, International Journal of Artificial Intelligence in Education 13, 2003, pp. 171-195, IOS Press.

[31] Gerald Kulm, Mathematics Assessment: What Works in the Classroom. San Francisco, Jossey-Bass, 1994, ISBN: 978-0-7879-0040-3.

[32] Chinmay Kulkarni and Scott R. Klemmer, “Learning design wisdom by augmenting physical studio critique with online self-assessment”, Technical report, Stanford University, July 2, 2012.

[33] John A. Ross, Carol Rolheiser, Anne Hogaboam-Gray, “Effect of self-evaluation on narrative writing”. Assessing Writing, 6(1), 1999, pp. 107-132.

[34] Edgar Codd, “A Relational Model of Data for Large Shared Data Banks”, In Communications of the ACM, Volume 13, Number 6, New York, NY, USA, 1970.

[35] Günter Vinek, Paul F. Rennert, A.Min Tjoa, *Datenmodellierung - Theorie und Praxis des Datenbankentwurfs* [engl. Data Modeling - Theory and Practice of Database Design], Physica, Heidelberg, 1982, ISBN 978-3-7908-0225-2.

[36] Georgios Evangelidis, Evangelos Pitsiougas, Dimitris Dervos, Martti Laiho, Fritz Laux, and José F. Aldana-Montes, “DBTechNet portal: A Gateway to Education and Training for the European Database Technology Professional”, in Proceedings of the eRA-4 International Conference on the Contribution of I.T. to Science, Technology, Society and Education, 24-26 September 2009, Spetses, Greece.

[37] Ismael Navas Delgado, Maria del Mar Roldán García, José J. Farfán-Leiva, Martti Laiho, Fritz Laux, Dimitris A. Dervos, and José F. Aldana Montes, “Innovative Unity in the Diversity of Information Management Skills Teaching and Training across Europe”, in IADAT Journal of Advanced Technology on Education, Vol. 3, No. 3, July 2009, pp. 407-445, ISSN: 1698-1073.

Composing Semantic Web Services Online and an Evaluation Framework

George Markou
Dept. of Applied Informatics
University of Macedonia
Thessaloniki, Greece
e-mail: gmarkou@uom.gr

Ioannis Refanidis
Dept. of Applied Informatics
University of Macedonia
Thessaloniki, Greece
e-mail: yrefanid@uom.gr

Abstract — **This article presents an approach for semantic web service composition using Artificial Intelligence planning techniques. Our research prototype, MADSWAN, is able to support various stages of web service composition, both manually and automatically. It comprises a semantic web service registry, an editor of semantic description files, as well as manual and automatic web service composition modules. The system adheres to the reusable nature of web services, by utilizing existing open source projects as sub-elements. Furthermore, it tackles the problem's inherent non-determinism through the use of planning techniques, particularly contingency planning. Finally, we designed a set of evaluation benchmarks for web service composition systems, based on an existing collection of semantically annotated web services, and applied it in part to MADSWAN with very encouraging results.**

Keywords - Web service composition, non-determinism, OWL-S.

I. INTRODUCTION

In [1], we presented our ideas in regard to a system aiming to automate web service composition procedures through the combination of semantic web technologies and Artificial Intelligence (AI) planning techniques. Here, we extend this work, presenting our research prototype, **MADSWAN** (an anagram of "**M**anual **AND** **A**utomatic **S**emantic **W**SC"), a general evaluation benchmark set for web service composition systems, as well as some experimental results concerning the efficiency and effectiveness of **MADSWAN**.

The main goal of the semantic web [2] is to offer unambiguous and computer interpretable markup of the web's content, as well as of its properties and relations, using a language that has explicit, well-defined semantics [3]. This makes it possible to automate tasks that could previously only be performed by humans.

Web services, a major ingredient of the Semantic Web, aim to solve interoperability problems between heterogeneous systems, with transparency over the underlying technologies used to implement them and the platforms they are based on. This aim is facilitated by the semantic markup of web services in a language such as OWL-S [4]. OWL-S presents what a service does, how it is used and the effects it has, thus enabling the automation of tasks such as web service discovery, selection and composition.

Since an atomic web service does not often provide the desired functionalities on its own, it is necessary to perform the task of Web Service Composition (WSC) in order to achieve them. However, web services exist and operate in an ever-changing and expanding environment. For that reason, searching for the appropriate web services to achieve each goal is not an easy task; what makes WSC difficult and time-consuming is the additional burden of manually monitoring whether a web service taking part in an existing solution is still active and has the same usage and interface.

The problem of automatic WSC has been shown to be computationally hard, most recently in [5, 6]; in [5], it is stated that if the WSC problem is formulated as a composition of finite state machines, each one representing a web service, then in its simplest setting, when the composition is fully asynchronous, there is an *EXP-hard* lower bound on the complexity of the task. In [6], it is shown that solving the composition problem of non-deterministic web services with complete information is *EXP-hard*.

In a deterministic setting, we are forced to assume that we can predict the results of the executed actions precisely. However, such a setting is often restrictive and not realistic, and the adoption of a non-deterministic assumption, i.e., that the outcome of a web service is not known a priori, allows us to compute more flexible plans. In our work, we assume that non-determinism is inherent in the WSC domain and that it is always possible for a web service's execution to be unsuccessful or have undesired effects.

The automatic WSC process includes the following phases: presentation (or advertisement) of a single service in a registry; translation between external and internal service specification languages for the domain; generation of a composition process model; and, finally, evaluation and execution of the output composite service [7]. **MADSWAN** currently supports the first four phases of the WSC process.

MADSWAN is based on open source software components that utilize the current web service standards, with the main goal of creating a platform that allows its quantitative evaluation and comparison to other WSC systems. A typical user can advertise a new web service in the online registry, as well as retrieve and edit the web services stored in it through the system's online interface. Moreover, it is possible to semi-automatically create workflows based on OWL-S control constructs and bind them to web service descriptions and concepts that are present in the online registry.

Most importantly, however, users are currently able to generate composition process models automatically, based on deterministic AI planning [8, 9]. AI planning is the task of coming up with a set of actions that will achieve a goal; our future goal is to support contingency planning [10]. Contingency planning anticipates the different outcomes of nondeterministic actions, plans for a subset or all the possible contingencies that could arise, and allows for the construction of a conditional plan that can be executed correctly despite those contingencies.

Following a translation of the original semantic WSC domain, described in OWL-S, to an AI planning one, defined in PPDDL [11], the translated problem is solved by an AI planner. The solution is then converted back to an OWL-S process model. The whole process can be evaluated against pre-defined use case scenarios and simple quantitative criteria, such as the number of atomic or composite web services included in the output composite one, as well as the total planning time required to reach a solution.

This article extends our previous work [1, 12] by providing a rigorous analysis of **MADSWAN**, presenting a working prototype, as well as focusing on its evaluation process. We provide details about the modules of the system, that is, the registry, the XML editor, and the manual and automatic WSC modules; we also provide benchmarks to evaluate **MADSWAN**, giving details with regard to the specific ontological concepts used by them. Finally, we evaluate the composite web service descriptions that are generated by **MADSWAN** for the deterministic setting against two different planners, using four problems from two different domains.

MADSWAN is, to the best of our knowledge, the first system of its kind with a publicly available prototype able to support various stages of the WSC process. In combination with the presentation of an algorithm aimed to tackle the non-determinism in the WSC domain and the provision of quantitative evaluation benchmarks, these constitute a unique set of features for such a system.

The rest of the article is organized as follows: Section II reviews related work; Section III presents technical details concerning the implementation of **MADSWAN**, the translation process between the WSC domain description language and the planning domain description language. Section IV focuses on evaluation by describing the benchmarks that are introduced to be used as test cases for WSC systems and presents an experimental evaluation of **MADSWAN.** It also showcases the manual WSC module that is used for the generation of workflows for the predefined scenarios. Section V concludes the article and poses directions for future work.

II. RELATED WORK

This section reviews the related work concerning, firstly the relevant WSC approaches, including those making use of AI planning and those that utilize different techniques and, secondly, the evaluation of such systems.

A. Web Service Composition

AI planning is the most widespread approach used to tackle the WSC problem. However, a significant number of approaches using different methodologies exist; although these approaches cannot be directly compared to the one presented in this article, we will briefly refer to them and then focus on the ones making use of AI planning.

An example of a non-AI planning approach is presented in [13]; its authors present a semi-automatic approach to WSC, with the output composite web services specified as process schemas, and the atomic web services that comprise the composite ones being selected at runtime, based on non-functional constraints specified by the users. The presented system, CCAP, is based on three core services: coordination, context and event services that schedule and implement user-configured adaptations of web services at runtime. The approach is considerably different than ours, since it is not fully automated and is only based on the syntactic content of web services. CCAP only makes use of technologies such as XML [14] and UDDI [15], without taking into account the semantic matching capabilities that can be achieved by using ontologies and semantic specifications, such as OWL-S.

The authors of [16] developed ITACA (Integrated Toolbox for the Automatic Composition and Adaptation of web services), a toolbox that supports the composition of BPEL [17] services in order to generate adaptation contract specifications. The process is based on the automatic extraction of behavioral models from interface descriptions that can be defined in WSDL [18], Abstract BPEL (ABPEL) or Windows Workflow Foundation (WF) [19]. However, as the authors note, although the adaptation process is automated, the final contract specifications may require human intervention to successfully complete the WSC process. Additionally, as in [13], the web services used do not carry any semantic content.

A framework for composing pre-existing services and components that is based on ITACA is presented in [20]. DAMASCo (Discovery, Adaptation and Monitoring of Context-Aware Services and Components) has been implemented as a set of tools that constitute a framework integrated in ITACA. The authors acknowledge the need for semantic representation instead of only a syntactic one and use model transformation, context-awareness, semantic matchmaking, dependency analysis and fault tolerance in order to achieve the goals of discovering, adapting and monitoring the composition of web services.

One of the first, and most well-known, approaches that convert the original WSC problem to a planning one is presented in [21]. The proposed system, SHOP2, converts the web services' OWL-S process models to a SHOP2 domain, and the WSC problem to a compatible Hierarchical Task Network (HTN). SHOP2 plans for tasks in the same order in which they will be executed, allowing it to be aware of the current state of the world at each step. Despite this advantage, the approach is planner dependent and does not deal with the domain's non-determinism Thus, it is limited in comparison to more general approaches that translate the WSC problem to one compliant with PDDL [22].

Another approach that utilizes AI planning to solve the WSC problem at hand is followed in [23], treating the application of a web service as a belief update operation. The authors identify two special cases of WSC that are tractable; these cases allow for a compilation into planning under uncertainty and the subsequent use of a conformant planner, that is, Conformant-FF [24]. This approach does not make use of a standardized web service description language, and the planner is only given as input a PDDL-like description of the domain, modified to describe uncertainty about the initial state.

PDDL and OWL-S are the de facto planning language and the most widely used semantic description language, respectively. For that reason, several attempts exist that tackle the WSC problem by utilizing the two languages in conjunction. OWLS-Xplan [25] is among the most well-known approaches utilizing a translation between PDDL and OWL-S. It incorporates a tool that translates OWL-S descriptions to corresponding PDDL-like ones, and then a hybrid planner is called to solve the generated planning problem, combining guided local search with graph planning and a simple form of HTN decomposition.

A similar approach is adopted in [26], in which standard PDDL files are produced during the translation phase, and consequently any PDDL-compliant planner can be employed to obtain a solution to the WSC problem. In practice, the authors incorporate two alternative planners, JPlan [27] and LPG-td [28]. A module that allows for the replacement of a single service if a user manually selects it for substitution is presented, with non-determinism in the WSC domain not being taken into account.

The authors of [29] also present a conversion schema from OWL-S to PDDL, based on [25, 30]. The presented methodology does not ignore the non-determinism in the domain, and makes use of a modification of an existing PDDL planner (Simplanner, [31]) to tackle it, through interleaving planning and execution. The proposed system does not support a full-featured registry that allows, e.g., the addition of new web services or the removal of existing ones by its users.

To the best of our knowledge, there are currently no web-based systems supporting multiples phases of the WSC process available. YaWSA [32] provides a web-based interface that supports a WSC process; however, it is no longer available for public use and, most importantly, it provided no other capabilities related to different phases of WSC, such as a registry.

The authors of [33] present a system supporting multiple phases of WSC, including web service browsing, the creation of composite services, service flow execution, and the generation of OWL-S descriptions used to describe their common process pattern instances. These instances are used to bridge the gap between the users' requirements and the technical service descriptions, as the authors view OWL-S as insufficient and not abstract enough to achieve such a result on its own.

The system that is most closely related to **MADSWAN** in terms of functionalities is the one implemented in the SUPER (Semantics Utilized for Process management within and between EnteRprises) project [34]. The major objective of SUPER was to bridge the gap between the business needs expressed by business people and the actual Information Technology (IT) infrastructures intended to support them, while also supporting in a more efficient way the reuse and automation of business processes. For this reason, it implements a semantic-based and context-aware framework platform that supports the management of business processes in a scalable manner, through the use of semantic web services' technologies.

The final platform includes modules for the automated discovery, substitution, composition and execution of business process implementations. Furthermore, three use case scenarios were developed for the needs of the project, all based on the telecoms domain, covering the fields of fixed telephony, traffic routing and the management of mobile environments.

Despite the different objective of SUPER project in comparison to **MADSWAN**, it shares a lot of similarities with it. The system's interface was alike the one in the manual composition module presented here, using the BPMN [35] standard as one of its basic elements. More importantly, the two systems share a similar architecture, e.g., the inclusion of modules for the discovery, translation and composition of semantic web services, even though the underlying standard for the description of semantic web services is WSMO [36] in SUPER and OWL-S in our case.

The evaluation process of SUPER was conducted solely based on interviews with a sample set of the system's users expressing their view on criteria such as the completeness and support, e.g., in terms of tools, of the system, or on its reuse of open source software and standards and its overall correctness. In our case, the evaluation process is based on quantitative criteria. The most important difference, however, lies in the WSC approach; WSC is used in SUPER to refine relevant parts of a business process model by searching for partial replacements in a process model. That is, the produced composite output is not presented as a new semantic web service, as in the approach presented here, nor does it take into account any non-determinism in the domain.

B. *Evaluation of Web Service Composition Systems*

It is noteworthy that the literature on WSC systems suggests a gap in their evaluation process; although recently there have been a few exceptions, a plethora of approaches simply rely on qualitative criteria and/or a single case study to evaluate their methodology. More importantly, the relevant literature does not suggest a standard test bed for WSC systems [23], or even a standard collection of web services to be used.

In [37], a comparison of planning techniques for WSC is conducted, with the evaluation criteria being based only on qualitative criteria. Some of the criteria that the authors take into consideration are: whether the technique is domain independent as well as whether it supports partial observability and non-determinism or not; the standards to which it can be applied to, i.e., its applicability, and its support of concurrency in the execution of web services. The scalability of the approach is also evaluated, but without any mention to quantitative results; the authors simply evaluate it based on a critique of the algorithm used.

In [38], current WSC approaches are criticized for making use of easy to measure criteria, without incorporating more important requirements in their evaluation. Several testing challenges in relation to the effective evaluation of service-centric systems are reported in [39]. Among these challenges are the dynamicity and adaptiveness that is inherent in the output workflows that contain abstract services, due to the fact that they can be automatically bound to various concrete services during the execution of the workflow instances. Other reported challenges are the lack of control that is attributed to a web service being modified during its lifecycle, and the cost of testing that is related to invoking the actual web services.

In [40], it is concluded that web service testing is more challenging than testing traditional systems; these findings are consistent with [38], that is, it is the authors' opinion that the difficulty in testing web services is partly due to the dynamic nature of web services and the limited control over them, as well as not having access to their source code.

Several approaches for WSC are evaluated in [41]. Among those not referenced previously, three out of eleven do not provide any evaluation at all. The rest, e.g., [42], present quantitative experimental results, such as the time needed to achieve a solution; however, none refers to actual scenarios with specific goals that exhibit the system's functionalities. For example, in [43], another one of the approaches evaluated in [41], randomly generated problems are used, with the authors only providing information regarding the number of services and ontology concepts present in them. The authors of [44] present quantitative experimental results, along with details in relation to the machine that was used to run the experiments and the number of web services taking part in the tests. A non-standardized language is used to implement the composition model, namely VCL (Vienna Composition Language), without any mention to the structure or the goals of the test problems, so as to allow their replication and comparison to other systems, or to testify that they are non-trivial.

Of the approaches evaluated in [41], only [45] denotes the test set that was used for evaluation purposes, the 2009 Web Service Challenge dataset [46]. The actual benchmark domains used in [47] are specified, consisting of two of the problem files used in [21].

Although a large part of the recent approaches related to planning, such as [29, 48], either evaluate their methodology on case studies without referring to quantitative criteria, or not at all [49], the same is partially true for non-AI planning approaches as well. This is the case for [16], where it is mentioned that the system has been validated and evaluated in synthetic problems and real-world examples, such as a travel agency, or library management systems, without, however, any results being reported.

Only recently a few AI planning approaches, such as [23, 26, 50], provided quantitative criteria for evaluation. The most extensive evaluation is presented in [23]; two artificial benchmarks are provided, each tested with different encoding methods and planners, and various elements of the planning process, such as the planner's total runtime or the number of search states and actions in the output plans, are measured.

In [24], a single case study is presented, with a different number of web services participating in the WSC experiments, and measuring the preprocessing, transformation (from OWL-S to PDDL), and planning time required. One of the two available planners in the system is used and evaluated, and the atomic web services that comprise the final composite one are (mostly) hand-tailored by the authors, although entire domains of the OWL-S Service Retrieval Test Collection (OWL-S TC) [51] are used for the composition in general.

Kona et al. [50] also incorporates rigorous evaluation experiments. Specifically, a single use case scenario is presented, with three variations depending on whether the produced workflows are sequential, non-sequential, or conditional non-sequential, along with the Inputs/ Outputs/Preconditions/Effects (IOPEs) of the services that take part in the WSC process. The criteria used in the evaluation of the system are quantitative, i.e., the number of web services participating in the problem, the number of I/O parameters each web service had, and the preprocessing and query execution time needed to obtain a solution. The web services that take part in the composition comprise a customized version of the 2006 Web Service Challenge [52] test collection.

An extensive evaluation is available in [20] based on two case studies; an online booking system and a road information one. Three variations for each are used, with increasing size and complexity with respect to the number of interface descriptions. The total numbers of states, as well as the required time and percent of CPU used are measured to evaluate the discovery and adaptation of both case studies.

Finally, the approach in [13] not only evaluates the system's performance based on scalability and adaptability criteria, but also provides a usability study based on 41 users of different educational backgrounds who were asked to use the system and report their experience by answering a questionnaire. However, the approaches in [13, 20] cannot be directly compared to **MADSWAN** mainly due to the systems' compatibility with dissimilar to ours underlying standards and technologies, such as the use of a UDDI registry in [13] with the services being described in WSDL, or the interface descriptions in [20] being defined either in WSDL, BPEL or WF. Secondly, due to their significantly different goals and motivation; for example, in [13] it is assumed that any non-determinism in the execution of web services has already been described in user configured exceptions.

III. THE MADSWAN SYSTEM

This section presents **MADSWAN**, particularly the modules that comprise it, their functionalities, as well as the steps of a typical use case, that is, creating a WSC problem, converting it to a planning one, solving it and, finally, translating the output plan to an OWL-S description file.

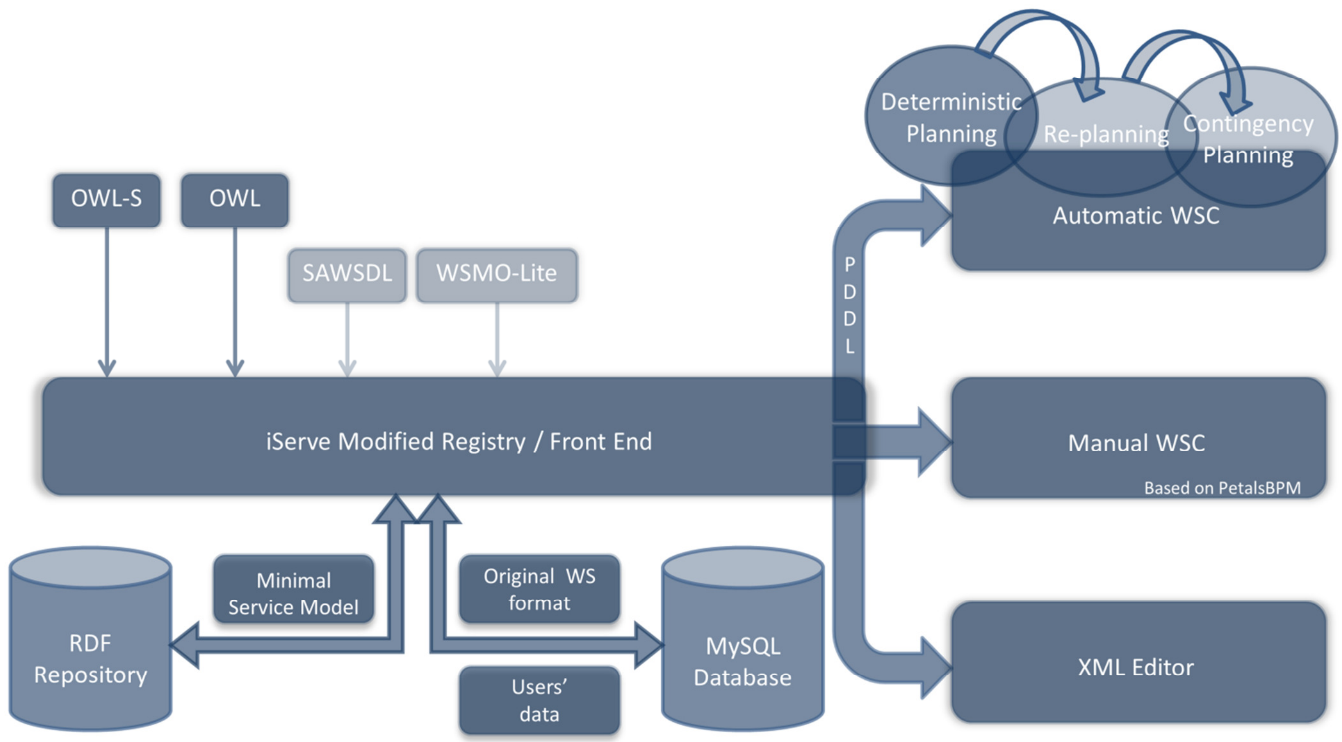


Figure 1. Architecture of **MadSwan**.

A. Available Functionalities

In our view, a WSC system should follow the same principles as web services themselves. Since web services rely on the idea of maximizing the reuse of loosely coupled components [53], our goal was to implement **MadSwan** by making use of freely available components as much as possible. This led to a reduced required effort in comparison to creating entirely new components, and allowed us to use well established standards instead of proprietary ones. Moreover, such an approach facilitates the comparison of different WSC systems to each other.

MadSwan supports various functionalities related to different stages of WSC, all of them being available through an online interface. An overview of the various components that comprise **MadSwan** is shown in Fig. 1. The core of the system communicates with an RDF repository and an SQL database, in order to store the users' data and the inputted semantic web services and ontologies. These are currently in OWL-S / OWL format respectively, with the prospect to support other formats, such as SAWSDL, in the future. The three different modules that correspond to the manual WSC, the automatic WSC and the online web service editor are depicted on the right of Fig. 1, with the transparent components depicting functionalities that have not yet been integrated to the system.

The first functionality related to WSC is storing the service descriptions. In order to support semantic web service discovery in a more meaningful way, we decided against the use of UDDI, although it is one of the most well-known approaches for web service publication. UDDI's search mechanism is based on the description of the web services' capabilities using a classification schema that does not provide for a semantic description of their content. For this reason, instead of using UDDI, or approaches such as [54, 55] that bridge the gap between semantic web services and UDDI (in most cases between OWL-S and UDDI), we opted to use iServe [56] as the core of our application.

iServe is a service registry that supports importing service annotations in various formalisms, such as SAWSDL [57], WSMO-Lite [58], and OWL-S. This process is achieved through first transforming the original annotations to linked data, based on a common vocabulary for services, called "Minimal Service Model". Since iServe is open source, we created a modified version of its web-based application, making several improvements to the original registry's interface and functionality, and populating it with version 4.0 of the OWL-S TC. Fig. 2 illustrates the basic functionalities of the implemented registry through screenshots of the actual application. It shows the sub-components that are available for a registered user of the system, with unregistered ones being allowed to access only a subset of them, namely the ones that do not alter the registry's contents.

Fig. 2 illustrates that the registered users can upload new web service descriptions, which in our case are semantically expressed in OWL-S, and search among the existing web service descriptions based on criteria such as their operation name, input or outputs parameters, or IDs. Moreover, they can view information regarding them, among which are their

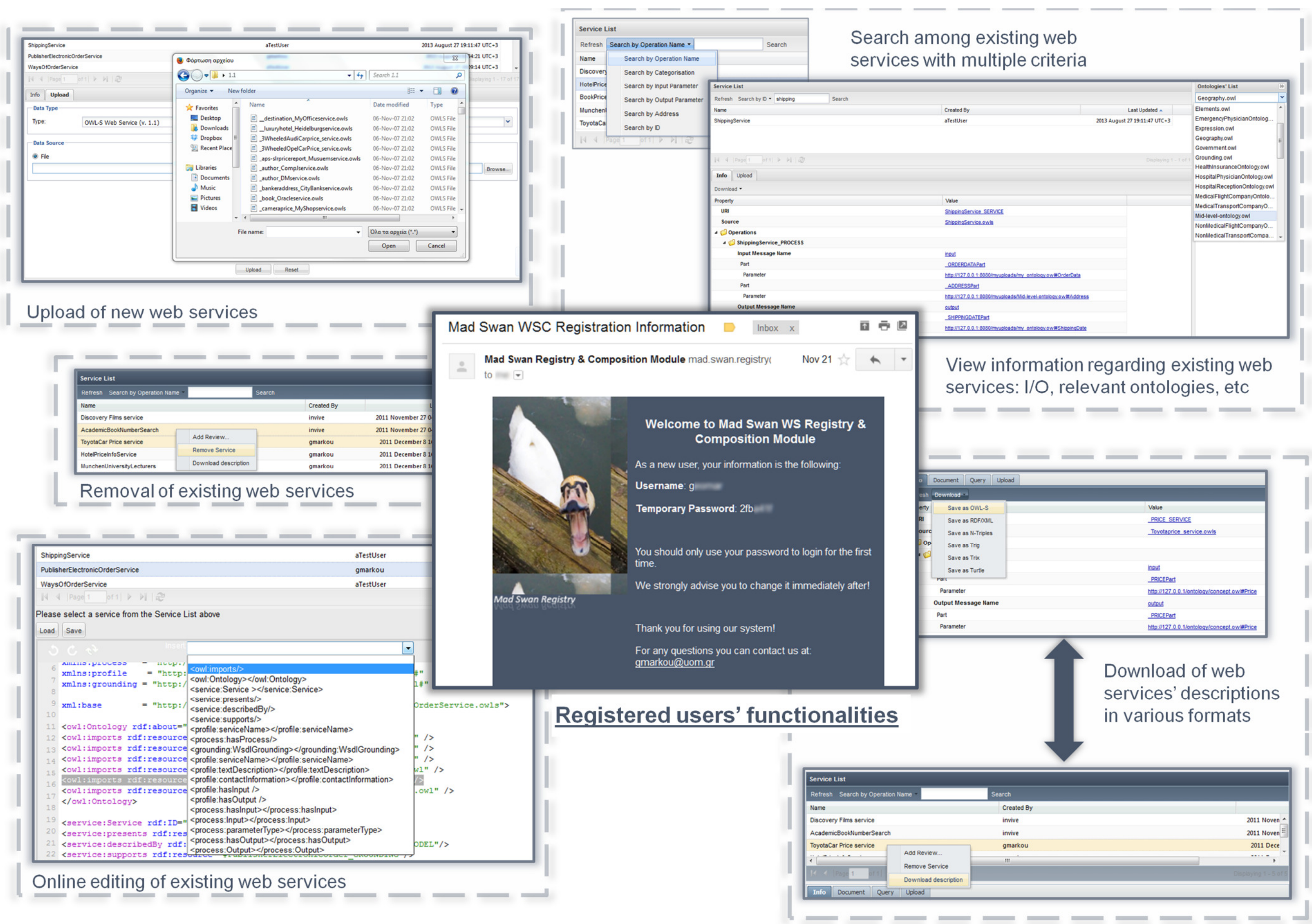


Figure 2. Registered users' functionalities for the registry – System screenshots.

Uniform Resource Identifiers (URIs), or their inputs and outputs based on the ontologies already in the registry. They can also remove or update the descriptions of existing web services stored in the registry, or download their descriptions either in their original format, i.e., OWL-S, or in a variety of other formats, such as RDF/XML [59] or Turtle [60].

Finally, the users have access to an online XML editor, which uses syntax coloring to facilitate its use, as well as predefined templates with OWL-S syntax, so as to allow its use by non-expert users. Fig. 3 overviews the interface of the application along with the XML editor module. Specifically, the four basic sub-modules of the application appear on the left side of Fig 3. On the top, there is the service list containing the available web service descriptions of the registry, along with their uploader's username and the date when they were last updated. This part of the interface also contains the system's search functionality. Directly below is the online XML editor, where a web service description from the registry is loaded, and the user is ready to insert one of the available template OWL-S syntax expressions. More information and tutorials on the basic use of the system are available at [61].

For the purposes of the non-deterministic automatic WSC, we plan to use PPDDL, the planning language used in the non-deterministic tracks of the recent International Planning Competitions for the purposes of the non-deterministic automatic WSC. PPDDL is essentially a syntactic extension of PDDL 2.1, and supports modeling non-deterministic actions through probabilistic effects, which can be arbitrarily interleaved with conditional effects and universal quantification.

Since the web services in the registry are described semantically through OWL-S, a translation between the two languages must take place. There are various works that have proposed conversion schemas from OWL-S to PDDL that do not differ significantly from each other. As such, we also adopt an approach similar to [25, 26, 30, 62].

Our translation module is based on the source code of [26], with the necessary extensions to accommodate the creation of planning files that can handle non-deterministic actions, that is, generating PPDDL files instead of PDDL ones, in cases where the domain is non-deterministic.

Moreover, the planning problem file is created based on the users' choices; a user can choose between the ontology concepts that are present in the web services that take part in

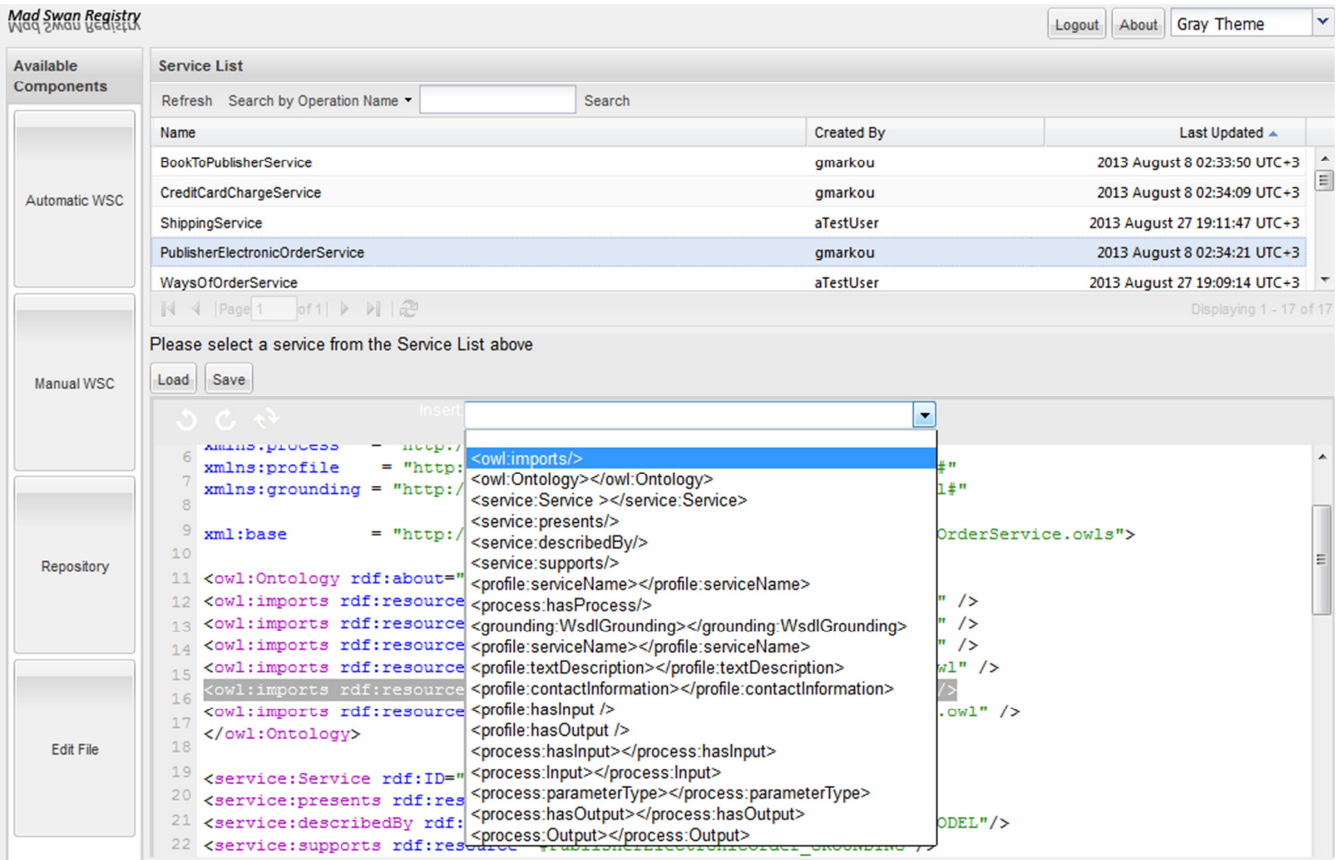


Figure 3. Application's interface presenting its XML editor.

the composition (either a subset of the web services in the registry that he selected, or all of them). Then he should state which concepts formulate the initial problem state and the final composite web service's inputs, and which formulate the goal state and the final composite web service's outputs.

The goal state concepts are split in two lists, one containing hard and one soft goals, essentially "must-achieve" and "should-achieve" ontological concepts. In the case that a user has stated a concept to belong in both lists, then the system considers that concept to be a hard goal. Fig. 4 presents the system's interface for such a case; the four different plans, with increasing lengths, that were generated are depicted, along with the total time that was required to produce them, the length of the plan with the minimum cost, and the total number of states that were expanded during the plan search.

Examples of the currently generated planning files are available at [63]; these files consist of the planning domain and problem files for a random problem, the solution file that the planner outputs, and the OWL-S profile and process files that are the results of the translation process of the solution plan. A set of semantic web service descriptions that can be used to interact with the registry (taken from OWL-S TC and modified accordingly) are available at [64]. Although the development of **MADSWAN** is still in progress, an alpha version of its online prototype is available at [65].

B. Contingent Planning

After the conversion of the OWL-S descriptions to planning domain and problem files, AI planning techniques can be used to generate the output plan/composite web service. In our view, the WSC problem is an inherently non-deterministic one. Indeed, it is always possible for a web service to be unavailable, or its execution to be unsuccessful or have undesired effects. For this reason, we adopt a non-deterministic formulation of the problem that allows us to compute more flexible plans. We opt for the incorporation of a contingent planner [66], in order to generate plans that can cope with the most influential and likely contingencies.

Our approach, which has not yet been integrated in the current version of the online prototype, is based on a complete search algorithm. This is used to generate all the possible plans for the most probable contingencies, starting from an optimal one, with an increasing cost, given a limited period of planning time. A - suboptimal - contingency plan can then be constructed by merging these plans. Merging is achieved through searching for natural join points, i.e., when search nodes share a predecessor through different sets of outcomes, and by removing any plans that contain redundant actions, that is, repetitive actions or ones that do not produce useful results.

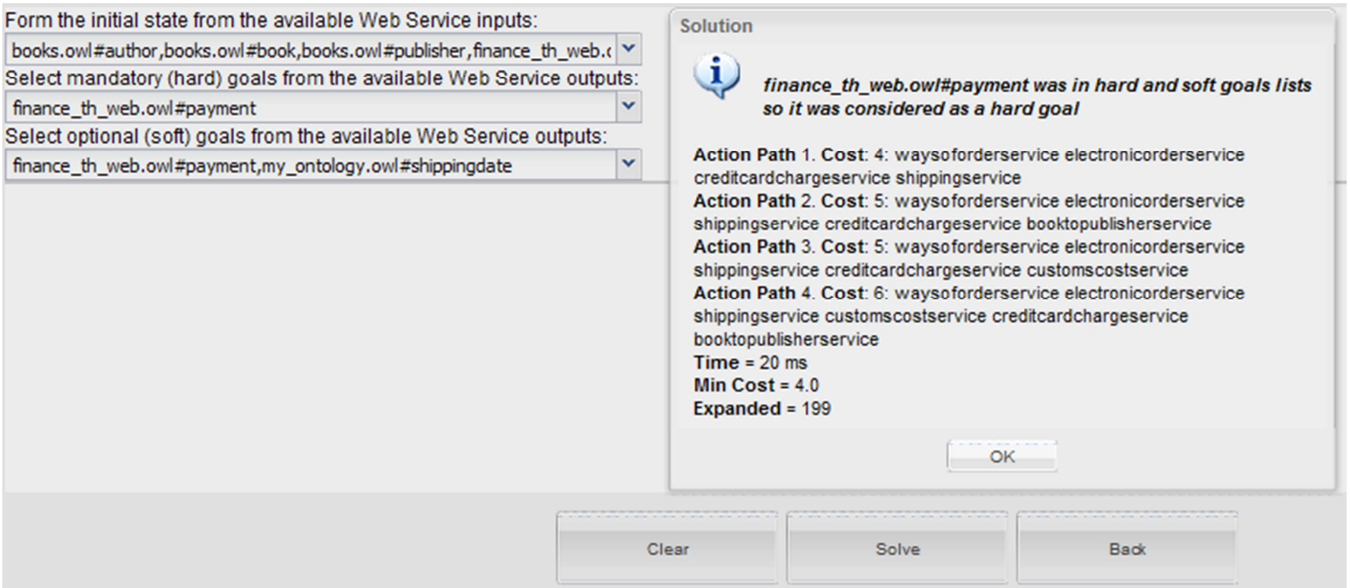


Figure 4. Application's interface regarding the automatic WSC module and its solutions.

A somewhat similar approach using GraphPlan [67] is presented in [68]. Here, we opt for the use of A^* [69], an optimal complete algorithm, with the use of the max heuristic (hmax, [70]) as an admissible heuristic function; however, this is not a restrictive choice. In practice, any complete algorithm can be used instead, in combination with a variety of admissible heuristic functions so as to produce optimal plans. The algorithm either outputs a contingent plan that covers all of the users' hard goals and any of the soft goals that have been set, or returns with a message that no plan was found in the allowed time period.

It is important to note that our approach does not try to develop a plan for every possible contingency, as the WSC domain may have too many sources of uncertainty for such a methodology to succeed. Since we cannot cope with every possible point of failure, a re-planning module will also be incorporated. As such, the approach is essentially offline, with a pre-computed contingent plan being used while the composite web service is executed. However, real time execution monitoring is essential, as the branches of the plan being used are determined by the actual outcomes of the atomic web services.

Re-planning occurs each time the contingent plan does not cover the current contingency, that is, an unexpected event occurs that is not already covered in the pre-computed plan. Such events may refer to a web service being unavailable at the time, or producing a result which is different than the one expected in the plan, e.g., a web service not being able to purchase a book due to it being out of stock.

Fig. 5 presents the aforementioned planning approach with an example. For reasons of brevity, we assume that the planning problem has three solutions. These solutions are of increasing length, the first being optimal with regard to the number of web services required to achieve the goal, requiring the execution of three actions (denoted by a rectangle), and the other two requiring an additional one.

The actions of the plans are color-coded, that is, the rectangles that share the same color are meant to depict the same action across different plans. We will refer to the actions by their respective color, e.g., the "green action". Each step in the creation of the plan is marked with its respective number on the right of the figure. Duplicate actions that are dropped from each plan and are not included in the final one are marked with a red "X".

In step 1, the contingent plan that we generate consists only of the base, optimal plan. In step 2, the second plan is added to the existing plan. Since the first two actions (white-purple ones) are common between the two plans, an alternative branch is created, consisting only of the two last actions (red-yellow). The new contingent plan is generated in step 3, and the next available deterministic plan is added to it in step 4. Since its first (white) and last (blue) actions are already present in the contingent plan, a new branch is added to it, consisting of the green and orange actions. The final contingent plan is shown in step 5, having three available paths to the goal, and for that reason, being able to withstand the occurrence of the same number of contingencies.

C. From AI Plan to Composite Web Service

Finally, we convert the plan back to an OWL-S (composite) web service, that is, we create an OWL-S profile and its process description, in a fashion similar to that described in [71]. In short, the resulting profile description file mainly refers to the new composite web service's IOPEs, as well as it defines some of its basic elements, such as its name or textual description. The output process model describes the plan that a client must execute in order to interact with the composite web service, and is based on OWL-S control constructs. The OWL-S API [72] that will be used to implement the conversion supports composite processes that use OWL-S control constructs, such as ⟨*Split+Join*⟩, and conditional constructs like ⟨*If-Then-Else*⟩, which is necessary to produce correct solutions to the use cases presented in Section IV.

Fig. 6 summarily illustrates our approach; it should be read as a timeline, starting from the left side of it. That is, OWL-S TC is used throughout all the stages of our application, whereas the OWL-S API is only used in the final stage, in the translation of the problem's solution to an OWL-S description file and the creation of the OWL-S profile and process files. Items in the same column imply that they are related to each other and occur in the same time frame.

D. Manual Web Service Composition Module

Since the relevant literature does not suggest a standard test bed for WSC systems, we decided to implement a manual WSC module so as to evaluate our automatic WSC approach against it. In order to create a manual OWL-S composer, we modified an existing open source application; Petals BPM [73] is a BPMN 2.0 modeler, which we adapted to accept the OWL-S constructs that are necessary for WSC. Moreover, we added a few "helper" constructs to provide a more intuitive interface.

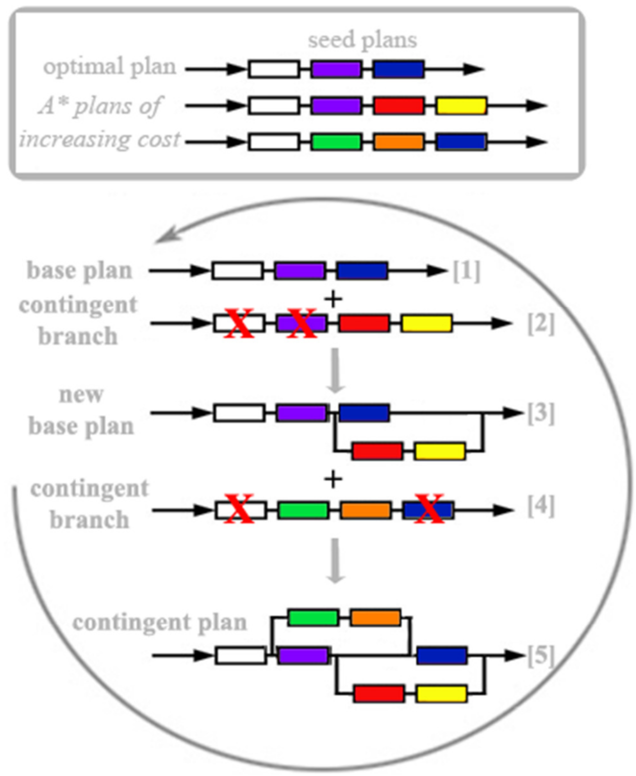


Figure 5. A graphic example of the planning algorithm.

The OWL-S constructs currently supported by the module are the ⟨*Sequence*⟩ (implicitly), ⟨*If-Then-Else*⟩, ⟨*Split+Join*⟩, and ⟨*Repeat-While*⟩ control constructs, along with the necessary inputs, outputs and web services' elements. The "helper" constructs comprise of an ⟨*End Split+Join*⟩ and an ⟨*End Repeat-While*⟩ construct, used

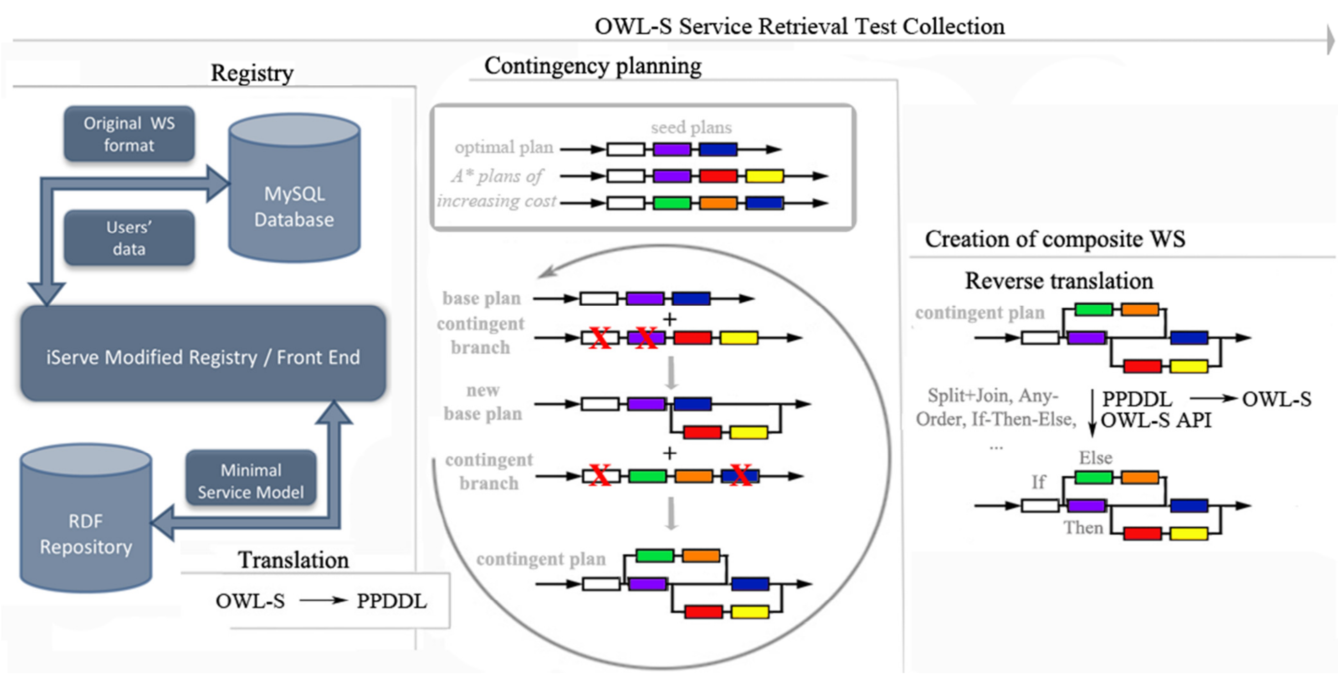


Figure 6. System overview timeline.

in conjunction with the regular ⟨*Split+Join*⟩ and ⟨*Repeat-While*⟩ constructs to enclose other elements in them, and dedicated ⟨*If*⟩ and ⟨*Else*⟩ sequence flows that are only used along with an ⟨*If-Then-Else*⟩ gateway. Moreover, there are ⟨*Start*⟩ and ⟨*End*⟩ constructs to signify the beginning and end of a workflow. Finally, users can bind the web service and data input/output constructs that they added to the workflows to specific web services and relevant ontologies' concepts already present in the registry, respectively.

It should be noted, however, that having bound such specific concepts from the registry to the available data input/output constructs, users are currently free to attach any ontology concept to any web service. That is, the system does not provide any semantic verification regarding the inputs/outputs of the web services, thus ensuring that a web service requiring a specific type of input is actually bound to one.

Since the purpose of the automatic WSC module is to help the non-expert, but familiar with WSC concepts, user to create composite web services, our aim was to implement the manual WSC module with the same principal in mind. For this reason, users can opt to be informed about the intended use of each construct available for the manual workflow creation, and the created graphical workflows are validated against pre-defined rules whenever the users save them.

Some of the pre-defined rules were maintained from the original application. An example of this is that the ⟨*End*⟩ construct, necessarily being the last one in a workflow, must have at least one incoming sequence flow from another construct. Other rules were added in order to help the user to export a valid composite web service, e.g., that an ⟨*If-Then-Else*⟩ construct is required to have an outgoing ⟨*If*⟩ sequence flow and can optionally have an ⟨*Else*⟩ sequence flow. The full list of added rules is presented in Table I.

TABLE I. OWL-S RULES REGARDING THE MANUAL WSC MODULE.

#	**Description:**
1	A workflow with a ⟨*Split+Join*⟩ control construct must also contain an ⟨*End Split+Join*⟩ one and vice versa
2	A workflow with a ⟨*Repeat-While*⟩ control construct must also contain an ⟨*End Repeat-While*⟩ one and vice versa
3	Any Data construct that is inserted must have its binding set
4	Any Web Service Task construct that is inserted must have its binding set
5	A Data Input cannot have any input connectors
6	A Data Input must have an outgoing connector
7	A Data Output can only be present following a Task or an ⟨*If-Then-Else*⟩ gateway if the gateway has a Task as its source
8	A Data Output must have an incoming connector
9	An ⟨*If*⟩ or ⟨*Else*⟩ Sequence Flow can only have an ⟨*If-Then-Else*⟩ gateway as its source
10	A Start Event cannot have incoming connectors except if they originate from a Data Input
11	Only Web Services'Tasks and their Data Outputs can be contained between a ⟨*Split+Join*⟩ control construct and an ⟨*End Split+Join*⟩ one
12	An ⟨*If-Then-Else*⟩ gateway must have exactly one outgoing ⟨*If*⟩ Sequence Flow and an optional ⟨*Else*⟩ Sequence Flow

IV. EVALUATION FRAMEWORK

This section provides details regarding three use case scenarios that can be used to evaluate WSC systems. It presents the graphical workflows that correspond to these scenarios through the use of manual WSC module. Finally, **MADSWAN** is compared to two existing planning approaches, in order to evaluate its efficiency and effectiveness.

A. Use Case Scenarios

As mentioned in Section II.B, no standard web service test bed or test collection exists. For this reason, it is currently very hard to evaluate a WSC approach objectively against another one, which is a detriment to the ongoing research regarding efficient WSC composition approaches.

One of our goals was to create an evaluation framework that could be used and reproduced by other systems, that is, clearly define detailed use case scenarios that are based on an existing, open test collection. We decided for the use of OWL-S TC, since in the past few years it has been used extensively, as a test set in the recent S3 contests [74], or in several approaches in the recent literature [25, 39, 75].

We have designed three use cases, each based on the service descriptions contained in a single domain of OWL-S TC, although several minor modifications were made to the descriptions of some services, and a few descriptions were added to some domains in order to design more useful scenarios. All the modifications to the original test collection, as well as a full description of the use cases, can be found in [76]. Next, we describe the use case scenarios in detail, with the ontologies' concepts used in each one shown in parentheses (following the format "*ontology#concept*").

Each use case scenario has an increasing amount of non-determinism and complexity compared to the previous one.

1) Movie Database Use Case Scenario: The first use case (**MADSWAN**-UseCase$_1$ - MS-UC$_1$) is fully deterministic, allowing for the output of a fully serialized composite web service; it refers to a user who knows the title of a film (*my_ontology.owl#Film*) and wants to retrieve all the comedy films (*my_ontology.owl#ComedyFilm*) that exist with a similar title, along with their respective prices. That is, he desires to know all the relevant pricing information in regard to a comedy film, i.e., its regular price (*concept.owl#Price*), its maximum price (concept.owl# MaxPrice), and its price with (*concept.owl#TaxedPrice*) and without taxes (concept.*owl#TaxFreePrice*).

Finally, the returned comedy films along with their pricing information results should be stored in a database (*ontosem.owl#database*), so that he can remember to buy them in the future. MS-UC$_1$ uses the web services in the "Communication" domain of the test collection, with the relevant ones in regard to the use case amounting to a total of 58 semantic web services.

The rest of the scenarios incorporate non-deterministic elements, such as alternative outcomes in the output composite web service based on the availability of items, or user preferences between different types of products.

2) Online Bookstore Use Case Scenario: The second use case (MS-UC_2) refers to a client of a specific online bookstore who wants to purchase a book; the client can use three different methods to buy the book, with alternative outcomes being outputted by the composite web service based on whether the book is in stock at the online bookstore or not.

In detail, the scenario describes a situation in which an online bookstore's client initially provides as input to the composite WS a book title (*books.owl#Title*) or its ISBN (*portal.owl#ISBN*), his address (*order.owl#Address*) and a preferred method of payment for the purchased item. The available choices for the customer to pay for the selected book are using his credit (*finance_th_web.owl#credit_card*), cheque (*finance_th_web.owl#cheque_card*), or debit (*finance _th_web.owl#debit_card*) cards. In order to suit the purposes of this test case, we slightly altered the *finance_th_web* ontology and added the debit and cheque cards as subclasses of cash, and the credit card concept as a subclass of credit.

As a result, if the book is in stock at the specific e-bookstore, the final composite WS should use the specified method of payment to purchase the item and record the address for the item in the user's shopping cart to be shipped to. The result should be a purchased item (*order.owl# PurchasedItems*), and the output of information regarding the purchased book; specifically, the book's author (*books.owl#Author*), along with its type (hard-cover or paperback) (*books.owl#Book-Type*), and size (small, medium or large) (*books.owl#Size*). However, if the book is not in stock, the client should not be charged with a fee and no information regarding the item should be displayed to him. MS-UC_2 uses the web services in the "Education" domain of OWL-S TC, with the relevant ones being 285 in total.

3) Camera Search Use Case Scenario: The final scenario (MS-UC_3) also concerns the purchase of an item; the main difference with MS-UC_2 is that multiple sellers can be considered for this scenario and, as such, the composite web service may need to check with all of them to determine the availability of the item. Moreover, MS-UC_2 and MS-UC_3 differ in that the latter also incorporates the user's preferences in the scenario. The user is assumed to have a preference, that is, soft goal, towards an analog SLR camera model; however, he may settle for other cameras if the preferred one is not available from any seller.

In specific, MS-UC_3 refers to a user who initially provides as input to the composite web service the type of the camera he desires to buy, which is an analog, non-APS, standard SLR camera (*extendedCamera.owl#SLR*). However, if this camera is not available, the scenario presumes that the user would also be satisfied with buying another type of analog camera that has less features, specifically an analog, non-APS, standard compact camera (*extendedCamera.owl#Compact*). In either case, the user also provides the desired camera's product code (*extended Camera.owl#ProductCode*). Finally, though, if no store is found having any of the two desired cameras in stock, then he will settle for any kind of camera at all, whether analog or digital (*extendedCamera. owl#Camera*).

The user is also expected to state in advance which stores should be considered as alternatives for him to buy the camera from; in detail, he can choose to provide as input a shopping mall (*Mid-level-ontology.owl#ShoppingMall*), a retail store (*Mid-level-ontology.owl#RetailStore*) or a specific chain of retail stores, with the available ones being Walmart (*Mid-level-ontology.owl#WalmartStore*) and Media Markt (*Mid-level-ontology.owl#MediaMarktStore).* Finally, he can also state that all mercantile organizations can be considered as alternatives (*SUMO.owl#Mercantile Organization*). Again, the user can provide one or more inputs, but in this case all the available choices are considered equally preferable alternatives. We assume that the user does not differentiate between the alternative stores he has provided as input, as long as he finds the camera he desires in stock.

Having entered the desired product type along with its product code and the alternative stores that can be used to buy it from, the composite WS should find a store that sells this product and check whether it has it in stock or not. If it is in stock, it should add it to the user's shopping cart (*ShoppingCart.owl#ShoppingCartRequestItems*); if not, it should continue to search for another store that sells it. If it cannot find any store that has the SLR camera in stock, it will repeat the aforementioned process, this time searching for a compact camera. If no compact camera is in stock either, then the composite WS will search for any camera available in stock. The output of the service can only be an addition to the user's shopping cart or no action at all. MS-UC_3 makes use of the test collection's "Economy" domain and of a total of 359 semantic web services.

We validate the correctness of the automatic WSC solution plan for a problem by checking that all of the ontological concepts that were present in the hard goals of its goal state have been generated by the web services in the plan. That is, the initial state concepts, along with the outputs of the web services that take part in the plan should be a superset of the hard goals set by the user.

This set of scenarios provides use cases that can efficiently evaluate the capabilities of WSC methodologies in a way that is both reproducible and extensible. They allow for a system to showcase that it can indeed cope with non-determinism in the WSC domain, and output both sequential and conditional plans, with and without taking into account the user's preferences.

B. Experimental Results

In order to test the correctness and efficiency of the current version of the automatic WSC module, we empirically compared a preliminary version of the algorithm presented in Section III.B against two existing planning

TABLE II. COMPARISON OF PLANNING TIMES FOR POND, LPG-TD AND MADSWAN.

	Opt	POND-EHC		POND-A*		POND-AO*		LPG-td		MADSWAN	
		Plan Length	*Time*	*Plan Length*	*Time*	*Plan Length*	*Time*	*Average Plan Length*	*Average Time*	*Plan Length*	*Time*
MS-UC$_1$	6	6	**0.0024**	6	0.0068	6	0.0032	6.04	0.0040	6	0.0169
P-10	6	6	0.0016	6	0.0020	6	**0.0016**	6	0.0066	6	0.0172
P-100	6	6	0.0036	6	0.0028	6	**0.0026**	6	0.0124	6	0.0117
P-1000	5	5	0.2640	5	2.5404	5	0.2742	5.76	0.1256	5	**0.0996**

a. The time measurements are in seconds; for the stochastic planner, LPG-td, they represent the median values of 100 runs. "Opt" represents the optimal length of each problem.

approaches; the first is a WSC system, PORSCE II [26], and the second a state-of-the-art contingent planner, POND [77].

We used two domains, one taken from the evaluation of [26], in order to provide direct comparison with PORSCE II, and one corresponding to MS-UC$_1$. The experiments were run on a PC using a Dual-Core Intel i5 processor running at 1.6GHz and allowing at most 4GB memory.

The domain taken from the PORSCE II system is deterministic in nature and, similarly to MS-UC$_2$, it models a user of a bookstore who wants to purchase a book. It is however a simplified version of MS-UC$_2$, as it does not feature non-determinism or any choices on behalf of the user. He is presumed to have a credit card that he can use to purchase a book, and the book is always considered to be in stock.

As in [26], we test three different versions of this domain, named *P-x*, "*P*" symbolizing the PORSCE system and *x* representing the number of web services participating in the problem. Since the web services are translated to PDDL actions, each version is increasingly more complex than the previous ones, the first consisting of 10 web services (*P-10*), the second of 100 (*P-100*) and the last of 1000 ones (*P-1000*). In the first two versions, the optimal plan length, i.e., the least amount of web services needed to achieve the desired goal, is 6, whereas the last version has an optimal plan length of 5. All of the aforementioned planning domains and problems are available at [78].

The second domain in the test set is from our own framework; since the proposed contingent algorithm has not yet been integrated in the online prototype, the use case scenario tested is the deterministic one, MS-UC$_1$. Specifically it is a relatively simpler version of it, with 21 web services taking part in the. The optimal plan length is also 6.

POND is a planner able to solve partially observable and/or non-deterministic problems by searching forward in the space of belief states, guided by a relaxed plan heuristic. It generates conformant and conditional plans using various search algorithms, specifically A*, AO* [79], LAO* [80], and Enforced Hill-Climbing (EHC) [81]. In our experimental setup we executed three versions of POND version 2.2, using the A*, AO* and EHC search algorithms.

As aforementioned in Section II, PORSCE II relies on two alternative planning systems, JPlan and LPG-td. For our experiments, we compare **MADSWAN** against LPG-td, as the authors of [26] concluded that its performance was by far superior to that of JPlan. LPG (Local search for Planning Graphs) is a sub-optimal anytime planner based on stochastic local search and planning graphs; its search space comprises action graphs, that is, particular subgraphs of the planning graph representing partial plans.

The experimental results are presented in Table II. In all cases, the time reported is in seconds and represents the total planner time needed to reach a solution. The planner with the best performance in each problem is highlighted in bold, and the optimal plan length (*Opt*) is shown next to each problem.

The experiments indicate that the number of web services available for WSC is a crucial factor in regard to the planner's efficiency, even if not all services are necessarily useful for the achievement of the goal. The results in Table II indicate that, generally, as the number of web services participating in each problem increases, so does the time required to solve it, although not linearly. This assumption is corroborated by the results of the comparison between MS-UC$_1$ and *P-10*, the two problems that comprise of just a few web services. MS-UC$_1$ requires more time than *P-10* for all versions of POND, and almost the same time for **MADSWAN,** but a little less time for LPG-td. This is not surprising, as the two problems share a common optimal plan length and MS-UC$_1$ comprises almost double the web services than P-10.

Moreover, although all the other problems in the experiments have a larger optimal length, *P-1000* appears to be by far the most difficult problem in the test set. The increase in the required time to solve *P-1000* for POND- A* is almost a thousand times more than for the second most difficult problem, *P-100*, a hundred times more for POND-EHC and POND-AO*, and tenfold for LPG-td and **MADSWAN**.

Although the number of web services comprising a particular problem is important, though, its difficulty is not dependent solely on this factor; e.g., **MADSWAN** manages to solve *P-100* faster than it does *P-10*. This fact can be mainly attributed to the complexity of the problem itself. The preprocessing phases of both LPG-td and all the versions of POND search for useless actions in the domain and allow the planners to ignore them during search (or simply prune them at parsing time). For this reason, both planners report that *P-10* consists of 10 (relevant) actions, whereas *P-100* of just 6.

Preprocessing leads to simpler domains, that is, for POND (that includes such facts in its output), *P-10* comprises 15 state variables while *P-100* comprises only 6. As such, both planners spend less time actually searching for a valid plan; on the other hand, though, this means that for relatively easy domains such as *P-100*, the vast majority of the total planner time is spent in the preprocessing phase. **MADSWAN** is more straightforward, devoting less time to preprocessing techniques; most of its total time is spent on search, allowing it to solve the – easier – *P-100* problem quicker than *P-10*, despite the fact that it searches among more web services/actions.

In general, the problems seem to be almost trivial for all planners, with the exception of *P-1000* for POND- A*, which has a significantly worse performance for this problem, both compared to the other planners and to its performance in the rest of the problems. All versions of POND, as well as **MADSWAN,** find solutions of optimal length for all problems. LPG-td, however, being a stochastic anytime non-optimal planner, returns plans of slightly worse median length for MS-UC$_1$ and (mainly) *P-1000*. **MADSWAN** needs more time than all versions of POND for the smaller problems (MS-UC$_1$, *P-10,* and *P-100*), as well as than LPG-td for the two smallest ones. On the other hand, it is faster than all planners for the most complex problem, *P-1000*.

It is important to note that since POND is not a WSC approach, but a standard PDDL planner, we experimentally tested only the efficiency of the planners on translated WSC domains to planning ones, and not on the whole process that **MADSWAN** typically follows. However, a major bottleneck of the solution of WSC problems seems to be the translation process from WSC domains to planning ones and vice versa.

Table III presents the results of our experiments regarding the average transformation time per web service description. In [26], it is reported that the average transformation time per web service converged to approximately 0.8 seconds. In our experiments, though, the necessary time was considerably less, converging to approximately 0.1 seconds per web service when translating a web service registry analogous to the size of the entire OWL-S TC.

Moreover, our experiments show a decrease in the average time required per web service translation as the total number of web services in the set increases, in contrast to the results reported in [26]. Since our translation process is based on the one in [26], we can assume that this fact, along with the significant improvement in the average transformation time per web service, can mainly be attributed to our different hardware setup, as well as minor optimizations in the original source code, mainly concerning the used data structures.

TABLE III. COMPARISON OF TRANSLATION TIMES PER WEB SERVICE FOR PORSCE II AND MADSWAN.

WSs	**PORSCE II**	**MADSWAN**
	Time	
10	0.4590	0.3974
21	-	0.2836
100	0.7000	0.1774
1000	0.7920	0.1110

a. All time measurements are in seconds and represent the median values of 100 runs. All the data for PORSCE II have been taken from [25].

The results of Table III indicate that most of the required computational effort for the automatic WSC process is attributed to the translation process of the original domain to the planning one, and not to the problem's solution itself. This is apparent by the fact that even when the average transformation time per web service converges to its lowest value (0.1110 seconds), the transformation time of a single web service is larger than the solution of a problem containing the same number of web services/actions (0.0996 seconds); that is, the time required by the planner to find a solution is less than one hundredth of the total time needed for the entire WSC process.

C. *Use Case Scenarios in the Manual Web Service Composition Module*

The existing implementation of the manual composition module is capable of producing graphical workflows that correspond to the three use case scenarios that were presented in Section IV-A. Fig. 7, 8 and 9 present the manually created workflows representing MS-UC$_1$, MS-UC$_2$ and MS-UC$_3$, respectively. In all figures, the labels of the connecting sequence flows have been removed so as to ease their legibility. The files used by the system's internal load and save functionalities for the three use case scenarios can be found at [82].

Using the automatic WSC module, the solution plan for the tests cases, e.g., MS-UC$_1$, requires just a fraction of a second; it is evident that even an experienced user would require at least a few minutes in order to accomplish a similar result visually in the manual WSC module. For that reason, the manual WSC module is mainly useful for the visualization of WSC workflows, and also for the creation of a basic "sketch" of a composite web service description, which can then be manually completed by experts.

We are currently working on the transformation of the created graphical workflows to their respective OWL-S descriptions and vice versa. The current functionalities of the manual WSC module are showcased in [65].

V. Conclusion and Future Work

In this article, we presented our work regarding the implementation of an online WSC system, named **MadSwan**, which makes use of AI planning techniques and of already freely available web service-related components. We provide a link to a demo of **MadSwan**, albeit still in alpha version, which showcases the system's functionalities.

This publicly available prototype is, to the best of our knowledge, the first online application of its kind able to support various stages of the WSC process. The fact that the presented system allows its users to store/retrieve web services to/from a registry, edit the ones already stored in it, and create new workflows both manually and automatically, all through an online interface, constitute a unique set of functionalities for such a system.

The current WSC approaches face several limitations, the most important of which is the plethora of available standards, both in relation to the semantic description of services and their underlying implementation. We acknowledge these limitations, which require substantial standardization efforts and coordination between the various service providers. For this reason, **MadSwan** is based on open source components, such as the manual WSC composition module and the registry, and utilizes existing datasets, i.e., OWL-S TC, and the current web service standards. This approach fortifies the web service principles the system is based on, and allows a more efficient comparison of the system to similar ones. Thus, we hope that our efforts constitute a step toward overcoming such incompatibility obstacles.

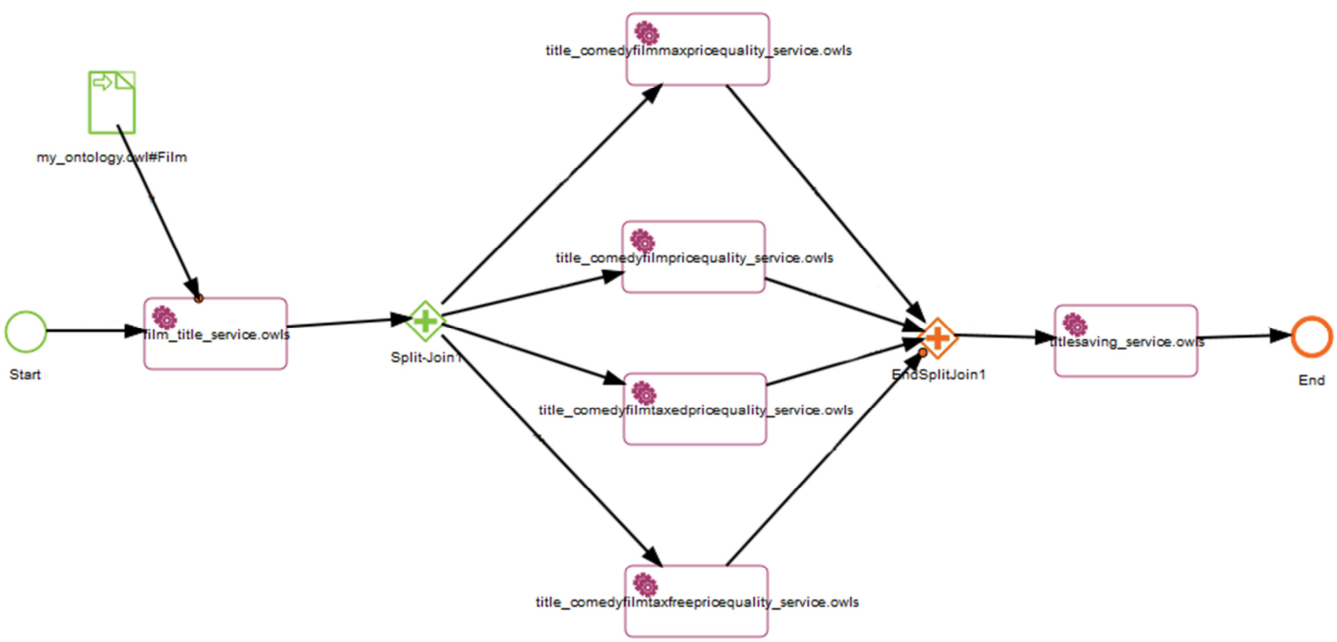


Figure 7. Movie database scenario workflow.

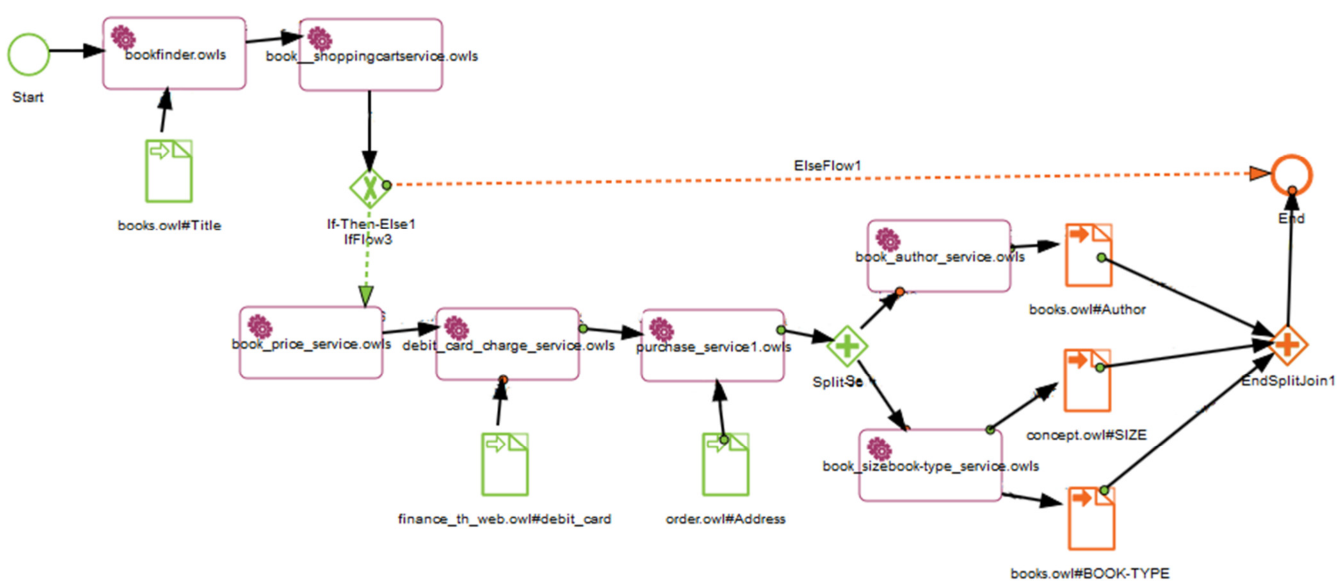


Figure 8. Online bookstore scenario workflow.

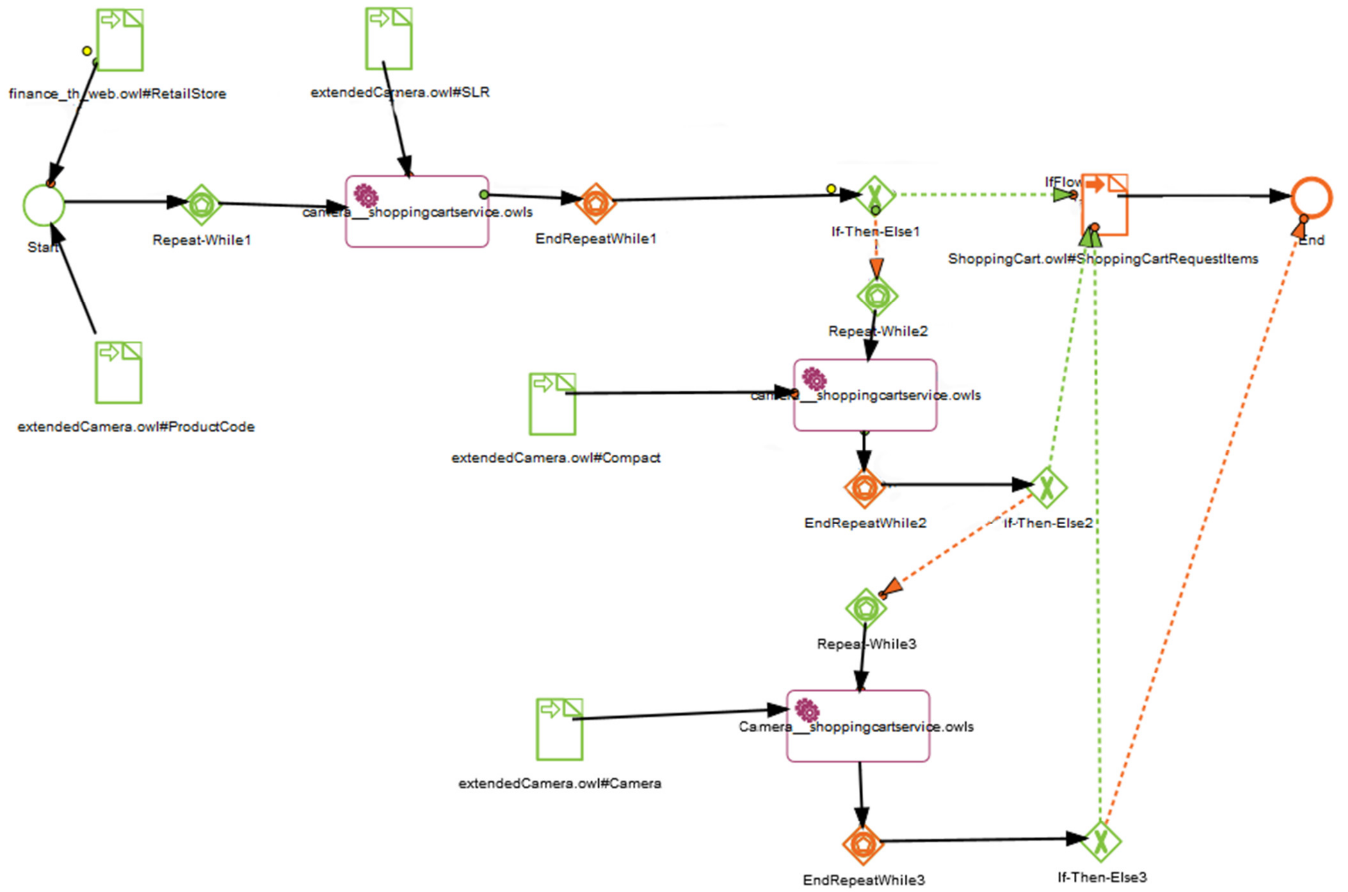


Figure 9. Camera search scenario workflow.

Currently, we are working on the design and implementation of a non-deterministic planning module, with the intention of finding plans that withstand various contingencies that can occur in non-deterministic domains. As demonstrated by our experiments, the current version of the prototype is able to find in a reasonable amount of time plans that solve deterministic WSC problems. The solutions are translated back to web service description files to be saved back to the registry for future use.

The article also presents in detail three use case scenarios that can be used to evaluate WSC processes. We established experimentally that the current, deterministic, version of the algorithm used in **MADSWAN** is competitive with both PORSCE II and POND in a variety of test cases. Furthermore, the efficiency of the translation process of the original WSC to a planning one was shown to be adequately fast for even a large repository of web services, e.g., one that contains all the web services in OWL-S TC.

Additionally, we provided implementation details in regard to the manual WSC module. This module can be used as an alternative counterpart to the automatic one; our goal, however, is that it will be used as a standalone application in order to help users create composite web service descriptions more efficiently compared to actually writing the entire description files by hand. Such a tool would allow users to familiarize with the concepts of web services and OWL-S, or simply to visualize composite OWL-S description files.

For the future, we plan to incorporate the contingent planning algorithm in the online prototype, and acquire experimental results for non-deterministic WS domains. We are also working on the semantic verification of inputs/outputs for the manual WSC module. Moreover, one of our long-term goals is to allow the execution of semantic web services, through the use of external tools such as SPEX (SPecification and EXecution tool) [83]. Finally, we aim to support automatic translation of arbitrary OWL-S composite web service description into the graphical workflows of the manual component.

ACKNOWLEDGMENTS

This research has been co-financed by the European Union (European Social Fund – ESF) and Greek national funds through the Operational Program “Education and Lifelong Learning” of the National Strategic Reference Framework (NSRF) - Research Funding Program: Heracleitus II. Investing in knowledge society through the European Social Fund.

The authors would like to thank Graham Crosmarie of the PetalsLink Research & Development Team for his time and advice in regard to Petals BPM, as well as the anonymous reviewers for their valuable comments.

REFERENCES

[1] G. Markou and I. Refanidis, "Towards Automatic Non-Deterministic Web Service Composition", Proc. *7th International Conference on Internet and Web Applications and Services* (ICIW'12), May 2012.

[2] T. Berners-Lee, J. Hendler, and O. Lassila, "The Semantic Web", *Sci. Am*., May 2001, pp. 29-37.

[3] S. Narayanan and S.A. McIlraith, "Simulation, verification and automated composition of web services", Proc. *11th International World Wide Web Conference* (WWW'02), May 2002, pp. 77-88.

[4] D. Martin, M. Burstein, J. Hobbs, et al. 2004. OWL-S: Semantic Markup for Web Services, http://www.w3.org/Submission/OWL-S/ [Dec. 12, 2013]

[5] A. Muscholl and I. Walukiewicz, "A Lower Bound on Web Services Composition", *Log. Meth. Comput. Sci.,* vol. 4 (2), 2008.

[6] W. Nam, H. Kil, and D. Lee, "On the Computational Complexity of Behavioral Description-Based Web Service Composition", *Theor. Comput. Sci.*, vol. 412 (48), 2011, pp. 6736-6749.

[7] J.Rao and X. Su, "A Survey of Automated Web Service Composition Methods", Proc. *1st international conference on Semantic Web Services and Web Process Composition* (SWSWPC '04), July 2004.

[8] S. J. Russell, and P. Norvig, "Planning", in *Artificial Intelligence: A Modern Approach*, 2nd ed., Upper Saddle River, NJ, Prentice Hall, 2003, ch. 11, pp. 375–459.

[9] M. Ghallab, D. S.Nau, and P. Traverso, *Automated Planning: Theory and Practice*, 1st ed., San Francisco, CA, Morgan Kaufmann, 2004.

[10] L. Pryor and G. Collins, "Planning for contingencies: A decision-based approach", *J. Artif. Intell. Res*, vol. 4, 1996, pp. 287-339.

[11] H. L. S. Younes and M. L. Littman, "*PPDDL1.0: An extension to PDDL for expressing planning domains with probabilistic effects*", Technical Report. School of Computer Science, Carnegie Mellon University, 2004.

[12] G. Markou and I. Refanidis, "Towards an Automatic Non-Deterministic Web Service Composition Platform", Proc. *8th International Conference on Next Generation Web Services Practices* (NWeSP '12), Nov. 2012.

[13] Q. Z. Sheng, B. Benatallah, Z. Maamar, and A. H. H. Ngu, "Configurable Composition and Adaptive Provisioning of Web Services", *IEEE Trans. Services Comput*., vol. 2(1), 2009, pp. 34-49.

[14] T. Bray, J.Paoli, C. M. Sperberg-McQueen, E. Maler, and F. Yergeau, "Extensible Markup Language (XML) 1.0", 2008. http://www.w3.org/TR/REC-xml/ [Dec. 12, 2013]

[15] UDDI version 3.0.2, http://uddi.org/pubs/uddi_v3.htm [Dec. 12, 2013]

[16] J. Camara, J. A. Martin, G. Salaun, et al., ITACA: An integrated toolbox for the automatic composition and adaptation of Web services. Proc. *31st International Conference on Software Engineering* (ICSE), May 2009, pp. 627-630.

[17] A. Alves,A. Arkin, S. Askary, et al., "Web Services Business Process Execution Language Version 2.0", 2007. http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html [Dec. 08, 2013]

[18] E. Christensen, F. Curbera, G. Meredith, and S.Weerawarana, "Web Services Description Language (WSDL) 1.1", 2001. http://www.w3.org/TR/wsdl [Dec. 12, 2013]

[19] Microsft Corporation, "Windows Workflow Foundation", 2007. http://msdn.microsoft.com/en-us/library/vstudio/ms735967(v=vs.90).aspx [Nov. 23, 2013]

[20] J. Cubo and E. Pimentel, DAMASCo: A Framework for the Automatic Composition of Component-Based and Service-Oriented Architectures. *The 5th European Conference on Software Architecture (ECSA),* Sept. 2011, pp. 388-404.

[21] E. Sirin, B. Parsia, D. Wu, J. Hendler, and D. Nau, "HTN planning for web service composition using SHOP2", *J. Web Semant.,* vol. 1, no. 4, Oct. 2004, pp. 377-396.

[22] M. Fox and D. Long, "PDDL2.1: An Extension to PDDL for Expressing Temporal Planning Domains". *J. Artif. Intell. Res*, vol. 20, Dec. 2003, pp. 61–124.

[23] J. Hoffmann, P. Bertoli, M. Helmert, and M. Pistore, "Message-based web service composition, integrity constraints, and planning under uncertainty: a new connection", *J. Artif. Intell. Res*, vol. 35, May 2009, pp.49-117.

[24] J. Hoffmann and R. Brafman, "Conformant Planning via Heuristic Forward Search: A New Approach", *Artif. Intell.*, vol. 170 (6-7), 2006, pp. 507 – 541.

[25] M. Klusch, A. Gerber, and M. Schmidt, "Semantic web service composition planning with OWLS-Xplan", Proc. *1st International AAAI Fall Symposium on Agents and the Semantic Web*, Nov. 2005.

[26] O. Hatzi, D. Vrakas, M. Nikolaidou, et al., "An integrated approach to automated semantic web service composition through planning", *IEEE Trans. Service Computing*, April 2011, pp. 301-308.

[27] JPlan: Java GraphPlan Implementation http://sourceforge.net/projects/jplan/ [Nov. 18, 2013]

[28] A. Gerevini, A. Saetti, and I. Serina, "An approach to temporal planning and scheduling in domains with predictable exogenous events", *J. Artif. Intell. Res*, vol. 25, Jan. 2006, pp.187-231.

[29] M. Kuzu and N. Cicekli, "Dynamic planning approach to automated web service composition", *J. Appl. Intell.*, vol. 36, January 2012, pp.1-28.

[30] H.S. Kim and I.C. Kim, "Mapping semantic web service descriptions to planning domain knowledge", Proc. *International Federation for Medical and Biological Engineering* (IFMBE), Aug. 2006.

[31] E. Onaindia, O. Sapena, L. Sebastia, and E. Marzal, "SimPlanner: An Execution-Monitoring System for Replanning in Dynamic Worlds", Proc. *10th Portuguese Conference on Artificial Intelligence* (EPIA 2001), Dec. 2001, pp. 393-400.

[32] A. Macdonald, "*Service composition with hyper-programming*". Technical Report, University of St Andrews, 2007.http://www.cs.st-andrews.ac.uk/~angus/docs/yawsa/final_report.pdf [Dec. 12, 2013]

[33] X. Du, W. Song, and M. Munro, "Using common process patterns for semantic web services composition", Proc. *15th International Conference on Information Systems Development* (ISD'06), Sept. 2006.

[34] SUPER - Semantics Utilized for Process management within and between EnteRprises, http://www.ip-super.org/ 08/30/2013

[35] Business Process Model and Notation (BPMN) Version 2.0, http://www.omg.org/spec/BPMN/2.0/ [Dec. 07, 2013]

[36] D. Roman, U. Keller, H. Lausen, et al., "Web Service Modeling Ontology", *Applied Ontology*, 2005, pp. 77-106.

[37] M. Chan, J. Bishop, and L. Baresi, "*Survey and comparison of planning techniques for web services composition*". Technical Report. University of Pretoria, 2007. http://polelo.cs.up.ac.za/papers/Chan-Bishop-Baresi.pdf [Dec. 12, 2013]

[38] R.W. Feenstra, M. Janssen, and R.W. Wagenaar, "Evaluating Web Service Composition Methods: The need for including Multi-Actor Elements", *Elect. J. of e-Gov. (EJEG)*, vol.5 (2), 2007, pp. 153-164.

[39] G. Canfora and M. Di Penta, "Testing services and service-centric systems: Challenges and opportunities", *IT Prof.*, vol. 8, 2006, pp. 10–17.

[40] M. Bozkurt, M. Harman, and Y. Hassoun, "Testing Web Services: A Survey", Technical Report, King's College London Centre for Research on Evolution, Search & Testing, TR-10-01, 2010.

[41] P. Bartalos and M. Bielikova, "Automatic Dynamic Web Service Composition: A Survey and Problem Formalization", Comput. Inform., vol. 30 (4), 2011, pp. 793-827.

[42] M. Alrifai, T. Risse, P.Dolog, and W. Nejdl, "A Scalable Approach for QoS-Based Web Service Selection", Proc. *1st International Workshop on Quality-of-Service Concerns in Service Oriented Architectures* (QoSCSOA '08), Dec. 2008, pp. 190-199.

[43] D.H. Shin and K.H. Lee, "An Automated Composition of Information Web Services based on Functional Semantics", Proc. *IEEE Workshop on Web Service Composition and Adaptation* (WSCA '07), July 2007.

[44] F. Rosenberg, M.B. Müller, P. Leitner, et al., "Metaheuristic Optimization of Large-Scale QoS-aware Service Compositions". Proc. *IEEE International Conference on Services Computing* (SCC '10), June 2010, pp. 97-104.

[45] W. Jiang, C. Zhang, Z. Huang, et al., "QSynth: A Tool for QoS-aware Automatic Service Composition", Proc. *8th IEEE International Conference on Web Services* (ICWS 2010), July 2010, pp. 42-49.

[46] Web Service Challenge 2009. http://ws-challenge.georgetown.edu/wsc09/ [Dec. 07, 2013]

[47] Y. Li, J. Huai, H. Sun, T. Deng, and H. Guo , "PASS: An Approach to Personalized Automated Service Composition", Proc. *IEEE International Conference on Services Computing* (SCC '08), July 2008, pp. 283-290.

[48] K. Chen, J. Xu, and S. Reiff-Marganiec, "Markov-HTN planning approach to enhance flexibility of automatic web service composition", Proc. *IEEE International Conference on Web Services* (ICWS'09), July 2009, pp. 9-16.

[49] Y. Bo and Q. Zheng, "A method of semantic web service composition based PDDL", Proc. *IEEE International Conference on Service-Oriented Computing and Applications* (SOCA '09), Dec. 2009, pp. 1-4.

[50] S. Kona, A. Bansal, M.B. Blake, and G. Gupta, "Generalized Semantics-Based Service Composition", Proc. *IEEE International Conference on Web Services (ICWS'08)*, Sept. 2008.

[51] SemWebCentral OWL-S Service Retrieval Test Collection, http://semwebcentral.org/frs/?group_id=89 [Dec. 01, 2013]

[52] M.B. Blake, W.K.W. Cheung, and A. Wombacher, "WSC-06: The Web Service Challenge", Proc. *8th IEEE International Conference on E-Commerce Technology and 3rd IEEE International Conference on Enterprise Computing, E-Commerce, and E-Services* (CEC/EEE'06), June 2006, pp. 422-423.

[53] J. McGovern, S.Tyagi,M. Stevens, and S. Mathew. *Java Web Services Architecture*. Morgan Kaufmann, 2003, pp. 35.

[54] N. Srinivasan, M. Paolucci, and K. Sycara, "Adding OWL-S to UDDI, implementation and throughput", Proc *1st International Workshop on Semantic Web Services and Web Process Composition* (SWSWPC '04), July 2004.

[55] J. Luo, B.E. Montrose, A. Kim, A. Khashnobish, and M. Kang, "Adding OWL-S support to the existing UDDI infrastructure". Proc. *IEEE International Conference on Web Services* (ICWS '06), Sept. 2006, pp. 153-162.

[56] C. Pedrinaci, D. Liu, M. Maleshkova, et al., "iServe: a linked services publishing platform", Proc. *Ontology Repositories and Editors for the Semantic Web Workshop at the 7th Extended Semantic Web Conference* (ORES '10), May 2010.

[57] J. Farrell and H. Lausen, "Semantic Annotations for WSDL and XML Schema", 2007. http://www.w3.org/TR/sawsdl/ [Nov. 23, 2013]

[58] D. Fensel, F. Fischer, J.Kopecký, et al., "WSMO-Lite: Lightweight Semantic Descriptions for Services on the Web", 2010. http://www.w3.org/Submission/2010/SUBM-WSMO-Lite-20100823/ [Dec. 09, 2013]

[59] D. Beckett, "RDF/XML Syntax Specification", 2004. http://www.w3.org/TR/rdf-syntax-grammar/ [Dec. 09, 2013]

[60] D. Beckett, T. Berners-Lee, E. Prud'hommeaux, and G.Carothers, "Turtle - Terse RDF Triple Language", 2013. http://www.w3.org/TR/turtle/ [Nov. 20, 2013]

[61] MADSWAN YouTube channel, http://goo.gl/kK5PAV [Dec. 12, 2013]

[62] P. Birchmeier, "*Semi-automated semantic web service composition planner supporting alternative plans synthesis and imprecise planning*". Diploma Thesis, University of Zurich, 2007. http://www.ifi.uzh.ch/pax/uploads/pdf/publication/1691/Birchmeier_Peter.pdf [Nov. 23, 2013]

[63] MADSWAN execution example files, http://ai.uom.gr/gmarkou/Files/IJAIT/ [Dec. 12, 2013]

[64] Sample semantic web service description files, http://ai.uom.gr/gmarkou/Files/OWL-S/ [Dec. 12, 2013]

[65] MADSWAN online prototype – alpha version, http://madswan.uom.gr/ [Dec. 10, 2013]

[66] J.Hoffmann and R.Brafman, "Contingent planning via heuristic forward search with implicit belief states". Proc *15th International Conference on Automated Planning and Scheduling* (ICAPS '05), June 2005, pp. 71-80.

[67] A. Blum and M. Furst, "Fast planning through planning graph analysis", *Artif. Intell.,*vol. 90, 1997, pp. 281–300.

[68] I. Little, "Paragraph: A graphplan based probabilistic planner", Proc. *5th International Planning Competition* (IPC-5), 2006.

[69] P. E. Hart, N. J. Nilsson, and B. Raphael, "A Formal Basis for the Heuristic Determination of Minimum Cost Paths", *IEEE Trans. Syst. Sci., Cybern.,* vol. 4 (2), 1968, pp.100–107.

[70] B. Bonet and H. Geffner, "Planning as heuristic search", *Artif. Intell*., vol. 129(1-2), 2001, pp. 5-33.

[71] E. Ziaka, D. Vrakas, and N. Bassiliades, "Translating web services composition plans to OWL-S descriptions", Proc. *3rd International Conference on Agents and Artificial Intelligence* (ICAART '11), Jan. 2011, pp. 167-176.

[72] OWL-S API introduction, http://on.cs.unibas.ch/owls-api/index.html [Dec. 06, 2013]

[73] Petals BPM architecture overview, https://research.petalslink.org/display/petalsbpm/Petals+BPM+-+Open+source+BPMN+2.0+modeler [Dec. 12, 2013]

[74] S3 Contest: Retrieval performance evaluation of matchmakers for semantic web services, http://www-ags.dfki.uni-sb.de/~klusch/s3/html/2011.html [Dec. 4, 2013]

[75] A. Mesmoudi, M. Mrissa, and M. Hacid, "combining configuration and query rewriting for web service composition", Proc. *IEEE International Conference on Web Services* (ICWS '11), July 2011, pp. 113-120.

[76] G. Markou, "*Heracleitus II - WSC Use Case Scenarios*". Technical report. University of Macedonia, 2012. Available at http://ai.uom.gr/gmarkou/Files/Mad_Swan_Use_Case_Scenarios.pdf [Dec. 12, 2013]

[77] D. Bryce, S. Kambhampati, and D.E. Smith, "Planning Graph Heuristics for Belief Space Search". *J. Artif. Intell. Res*, vol. 26, May 2006, pp. 35–99.

[78] Evaluation Domain and Problems, http://ai.uom.gr/gmarkou/Files/EvaluationDomains/ 12/012/2013

[79] N. Nilsson, *Principles of Artificial Intelligence*, San Francisco, CA, Morgan Kaufmann, 1980.

[80] E. Hansen and S. Zilberstein, “LAO: A heuristic-search algorithm that finds solutions with loops”, *Artif. Intell.*, vol. 129 (1-2), 2001, pp. 35 – 62.

[81] J. Hoffmann and B. Nebel, “The FF planning system: Fast plan generation through heuristic search”. *J. Artif. Intell. Res*, vol. 14, Jan. 2001, pp. 253–302.

[82] Use case scenarios’ output files screenshots, http://ai.uom.gr/gmarkou/Files/IJAIT/Scenarios(BPMNandScreenshots).rar [Dec. 12, 2013]

[83] J.T.E. Timm and G.C. Gannod, “Grounding and Execution of OWL-S Based Semantic Web Services”, Proc. *IEEE International Conference on Services Computing* (SCC ‘08), July 2008, pp. 588-592.

The Forwarding on Gates Architecture: Flexible Placement of QoS Functions and States in Inter-Networks

Florian Liers, Thomas Volkert, and Andreas Mitschele-Thiel
Integrated Communication Systems Group
Technical University of Ilmenau
Ilmenau, Germany
e-mail: {Florian.Liers, Thomas.Volkert, Mitsch}@tu-ilmenau.de

***Abstract*—A main driver of Future Internet applications and services is Quality of Service (QoS). Current Internet technologies provide no suitable QoS support for end-to-end connections due to several drawbacks of IntServ and DiffServ. In this article, we propose the "Forwarding on Gates" (FoG) architecture, which answers the QoS questions by the help of a new inter-network architecture. It applies its own new network protocol, which was designed to handle IntServ and DiffServ in an integrated way. FoG supports resource reservations for QoS guarantees in IntServ scenarios and prioritized traffic in DiffServ scenarios as well as a combination of both. As core advantage, the QoS support of FoG works in a scalable way by allowing a network to move QoS states and delegate decisions about the QoS usage to the entities demanding for QoS. This article describes the architecture, its network protocol, and solutions for interoperability with current networks. The evaluation includes theoretical descriptions of network configurations for a use case not supported by IP. Moreover, simulations show that the protocol overhead is comparable to IPv6, although packets can select QoS explicitly. Measured routing graph sizes for various setups show the flexibility of the FoG architecture.**

Keywords—Future Internet; network protocol; architecture; QoS; routing graph; interoperability.

I. INTRODUCTION

The Future Internet will be faced with much more applications requiring Quality of Service (QoS) than today's networks. In the broad field of Future Internet research, we focus on the applied protocol of the network layer. In [1], we started to describe an inter-network architecture that supports QoS support in a flexible and scalable way. In this article, we give more details about our inter-network architecture "Forwarding on Gates" (FoG) such as its incremental routing process and possible interoperability with IP. Moreover, we present recent evaluation results for its routing.

Applications requiring QoS support will be the main driver for protocols in future networks. The most important application is video streaming, which already stresses today's networks, especially the Internet. Forecasts predict that in 2015 about 62% of the traffic in the Internet will be video data [2]. For live video streams, as required for remote medical operations and for football games, QoS is required. Due to the large number of hosts and connections in the Internet, scalability is the crux for QoS.

For IP (both version 4 and 6), add-ons called IntServ and DiffServ have been developed in order to tackle QoS requirements in the Internet. However, both have pros and cons. The IntServ approach [3], with its signaling protocol "Resource Reservation Protocol" (RSVP), provides end-to-end QoS by introducing states on each intermediate router, which is passed by a flow. According to [4], RSVP is used to distribute states for classification, scheduling and reservation. The classification state defines how incoming packets are mapped to flows. With RSVP, such a mapping consists of a source address, a destination address and a protocol number (and optionally port numbers). A scheduling state defines how flows are handled. For example, on a node a flow can be assigned to an own prioritized packet queue for the outgoing hardware interface. Finally, signaling states represent management information, e.g., authentication data and timers. For each flow, an intermediate node requires one set of the described states. Due to memory limitations, such an approach causes scalability problems for scenarios with many flows [5].

DiffServ [6] was developed with main focus on scalability. It introduces a small set of QoS classes, which are used inside networks. Each QoS class defines a type of service and requires scheduling and signaling states. Thus, in comparison to IntServ, the number of states does not depend on the number of connections. However, DiffServ is not able to provide guarantees, since it is not aware of each individual flow. For a DiffServ network, edge routers of a network store and handle the classification states in order to map incoming packets to network internal QoS classes. The classification states represent the rules for this mapping. Since most interfaces with incoming traffic transport multiplexed flows, e.g., multiple TCP connections over the same Ethernet link, the classification is mainly done by (more or less deep) packet inspection. For example, protocol numbers, port numbers and even packet sizes are used for such classifications.

Besides their single application, IntServ and DiffServ can also be used in a combined way in order to leverage the advantages of both approaches. IntServ provides the signaling for flows between ingress routers and DiffServ provides a set of QoS classes used inside networks [7, 8]. However, the scalability problem of IntServ now appears at the ingress routers. They have to store the classification states per flow. Since the amount of scheduling and signaling states remains limited due to the limitation of possible DiffServ classes, the distribution of classification states is the main problem. In order to maintain the states, signaling is required. The processing load of handling these messages increases the burden on a network. In the past, proposals focused on reducing the number of flows, e.g., through aggregation [4].

Our key contribution is the proposal of an orthogonal strategy: move the classification states away from ingress routers to routers handling smaller amounts of flows. Furthermore, some decision-making authority is delegated from the QoS provider to the entity using QoS in order to reduce the required signaling overhead. As discussed in more detail in Section IV, today's network protocol IP is not able to support both in all required use cases. Therefore, we focus on a network protocol enabling the movement of classification states and the delegation of decisions between routers. Our solution is suitable both for IntServ and DiffServ scenarios. It is further able to handle combinations of both. Its main feature is the flexible placement of the classification states according to the network graph and the load in the system. It enables the handling of both QoS approaches in a single mechanism.

The remainder of this article is structured as follows: Section II describes our system architecture. Section III introduces the protocol and how its header is processed. Afterwards, in Section IV, the implementation of the use cases based on our architecture and protocol are presented. Subsequently, Section V shows possibilities for interoperability with current networks. Afterwards, Section VI shows our recent evaluation results from protocol simulations and Section VII shows how FoG is classified within related work. In the end, the main results of this work are summarized and an outlook about future steps is given.

II. Forwarding on Gates Architecture

As fundamental design aspect, the "Forwarding on Gates" (FoG) architecture separates the forwarding from the routing. It splits them into two logical components, which encapsulate their specific tasks. The forwarding component is responsible for relaying packets between routers and hosts. It handles the resource management and enforcement of resource reservations in order to take non-functional properties such as delay and bandwidth into account. The routing component is responsible for calculating paths through the network with respect to non-functional requirements given by applications [9]. Both are linked via a route definition. The routing component specifies a route and the forwarding component forwards packets along this route. The authentication component is the third logical component of FoG. It checks the authentication of the sender of signaling messages in order to secure access to management functions. This authentication component is the basis for authorization decisions and accounting for QoS provisioning. Figure 1 depicts these components and their interactions.

For the following discussion, we introduce the term *QoS function*, which generalizes QoS provisioning regardless of the underlying QoS architecture. A QoS function represents the setup, which is required to send packets with QoS constraints. Examples of QoS functions are setups implementing a DiffServ class or an IntServ reservation. QoS functions can provide guarantees ranging from "hard", with fixed limits, over "soft", with probabilistic QoS guarantees, to vague goals, e.g., "optimized for delay" or "best-effort". A QoS function comprises its scheduling and signaling states. The classification states are not included.

In addition to the separation of routing and forwarding [10], our architecture has some more features. It reduces forwarding table sizes [11], enables routers to choose their address format [12], hides addresses from applications, and supports various intra-network techniques. However, they are shared with other approaches from related work (see references) and are not in the focus of this article.

FoG applications specify their QoS requirements explicitly via an interface and FoG reacts accordingly. The interface is based on the "G-Lab Application-to-Network Interface" (GAPI) defined in [9].

A. Forwarding component

Today's Internet operates over interfaces of routers and hosts and links in between. FoG's forwarding component uses a virtual representation of the network in the form of a graph. Hosts and routers are represented by one or more vertices, which are called *forwarding nodes* (FNs). Edges between forwarding nodes are called *gates* and represent uni-directional (virtual) links between them. In order to support QoS, multiple edges between adjacent nodes are allowed. An edge is equivalent to a link with a QoS function between two routers or hosts. Each outgoing gate of a forwarding node is assigned to a *gate number*. The gate number is always unique to the scope of the forwarding node, to which the gate belongs to. Each FoG packet includes a header, which describes explicitly the order of gates the packet has to pass.

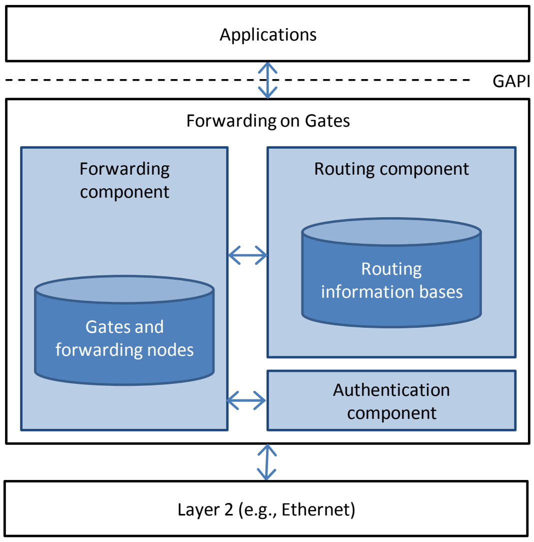


Figure 1: Logical components of the FoG architecture

Details about such a route, and how a forwarding node processes it, are given in Section III.

Gates are set up with a FoG-specific management protocol. The forwarding nodes process the signaling messages of that protocol and modify the graph of forwarding nodes and gates as requested. In order to secure this management, signaling messages are signed via the authentication service by the sender. The receiver uses the authentication service to verify the signature again.

The forwarding component informs the routing component about available gates and forwarding nodes to enable route calculations based on this information. However, gates that are not intended for other data flows (either connection based or connectionless) can be hidden in order to exclude them from the routing calculations.

B. Routing component

The interaction between the forwarding and routing is most important for FoG. Routes define how both are interacting. Whenever a route ends and the destination is not reached, the forwarding has to contact the routing component for the next (partial) route leading closer to the destination.

FoG routes are defined as stacks of segments. Two types of segments are possible:

- Explicit route segment: a stack of gate numbers
- Destination segment: the name or address of the destination, if a route is a partial one. The format and length depend on the routing and might range from DNS names to IPv4/6 addresses. Additionally, the QoS requirements are included. They describe the desired application requirements, which have to be considered by further route calculations.

A forwarding node uses the topmost segment of the route for its forwarding decision. If it is an explicit route segment, the forwarding node extracts the topmost gate number. This number is used for the following lookup for the next outgoing gate. In the simplest implementation, the gate number is an index representing the position of the outgoing gate in a stored gate vector. The packet passes this outgoing gate with a route, which is shortened by the used gate number. If the route segment is empty, it is removed from the route and the forwarding node proceeds with the next segment. If this is a destination segment, the forwarding node has to contact the routing component in order to know the next partial route towards the packet destination. For such a request, the forwarding node itself is the starting point for the route calculation. In general, the architecture does not restrict the format of the name or address, which are used in order to describe the packet destination. An implementation of the routing component can operate with its own format. If the destination segment includes QoS requirements, they are also included in the route request. Otherwise, best-effort routing based on hop costs is provided. The request result from the routing component is added to the route in the packet header by the forwarding component and the forwarding node will re-start its forwarding procedure once again. If there are no more route segments, the packet has reached its destination.

Destination segments can also be combined. In such a case, a destination segment is not necessarily the last segment in a route. It might define only the ingress router to an AS and a subsequent explicit route segment defines the rest of the route to the destination host. Theoretically, multiple destination segments in one route are possible. This allows for an *incremental routing* process, which is comparable to loose source routing of IP. Due to security considerations [13], policies can limit the utilization of this routing method. Since such limitations are mainly important at AS borders, multiple destination segments could be used within intra-networks. For example, this could be used to select the route through a local network.

In general, the subset of a network that is known to a routing component is a connected graph, which represents the nodes and links of the network itself and its surroundings. The knowledge about the parts outside of this subset is more abstract. A routing component knows about the connectivity but does not know the gate numbers, which are required to specify the route explicitly. Such abstract connectivity information can be represented by gates without gate numbers. They represent connectivity enabled by an unknown set of forwarding nodes and gates. This is comparable with the situation known from the Border Gateway Protocol (BGP). A BGP entity knows about the existence of a route (and its cost) but does not know the outgoing ports of each distant intermediate router, which has to be used in order to reach the destination.

If there are insufficient gates to form an end-to-end route, a routing component can contact the forwarding component and request the setup of new gates, depending on the physical connectivity. In particular, routing components can request gates with specific QoS capabilities in order to satisfy the QoS requirements of a particular route request.

The routing component can be implemented in multiple ways and with various protocols. We have developed a hierarchical approach for the implementation of the FoG routing component. It supports QoS and is called "Hierarchical Routing Management" (HRM). It clusters proactively the network at different hierarchical levels where each cluster has its own coordinator instance. The higher such a coordinator is located in the hierarchy, the more abstract is its network topology view. HRM uses these coordinators for distributing aggregated topology information among the physical nodes. As a result of this, each node knows the next hop for every existing destination without having global knowledge about the entire network topology. Further details about the approach are described in [14]. Additionally, we have already showed [15] the qualitative advantages of HRM for QoS demanding applications.

C. Authentication component

The authentication component is used to generate signatures for signaling messages and to check such signatures. Based on the authentication check, authorization decisions and accounting are done. It is mainly used by the forwarding component to secure the gate management. Furthermore, applications can use the authentication in order to sign data packets. Thus, the authentication has an impact on the packet structure described in the next section. However, due to the QoS focus of this article, the authentication component is not described in more detail.

III. FoG PACKETS

The FoG packet structure is shown in Figure 2. It starts with a header comprising all information required to decide the next hop. This enables a router to make a forwarding decision before the packet is fully received. The packet ends with a trailer containing all information that can be added to a packet after receiving it completely. The trailer is optional and can be omitted. The header and trailer elements are defined as follows:

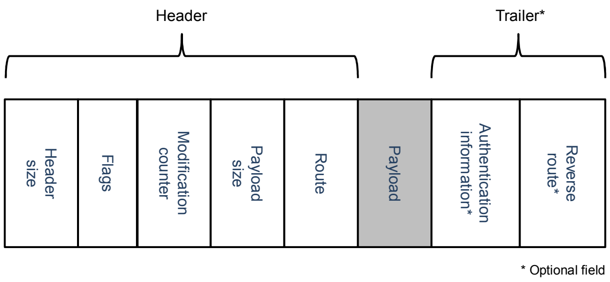


Figure 2: FoG protocol packet structure

- Header
  - *Header size in bytes:* This field is required to access the payload as subsequent versions of the FoG protocol specification could add more fields to the protocol header and before the payload.
  - *Flags indicate:*
    - if the reverse route in the trailer is present,
    - if the packet is a signaling message, and
    - if the authentication information in the trailer is present.
  - *Modification counter*, which is used to avoid infinite routing loops due to invalid gate setups
  - *Payload size* in bytes
  - *Route* for the packet

- Trailer
  - *Authentication information* (including a size field, which stores the length of the authentication information in bytes)
  - *Reverse route* for answers: This field includes the reverse route as it was recorded by already passed forwarding nodes.

Each route starts with a length field followed by a stack of route segments. Each stack entry contains the following fields:

- Segment type indication, where two types are allowed:
  1. An *explicit route segment*: This is a stack of gate numbers, defining explicitly a sequence of gates, which have to be passed by the packet.
  2. A *destination segment*: It contains the name or the address of the desired destination, and the requirements for the remaining route to this destination.
- Segment length in bytes
- Segment content with variable size: It contains the gate numbers of an explicit route segment or an address plus requirements of a destination route segment.

Each forwarding node processes the route of a packet as shown in Figure 3. If the route is empty, the packet has reached its destination. If the signaling flag is set, the packet contains signaling information dealing with the setup of gates and connection establishment between applications. The signaling message is handled by the host or router, and it updates the gate and forwarding node graph accordingly. If the signaling flag is not set and the forwarding node has an attached socket to an application, it removes the FoG header and trailer and stores the payload in the receive buffer of the socket. If the route is not empty, the forwarding node processes the topmost segment. If it is an explicit route segment, it removes the topmost gate number and uses it as locally unique identification for addressing one of its outgoing gates. If there is an outgoing gate with this gate

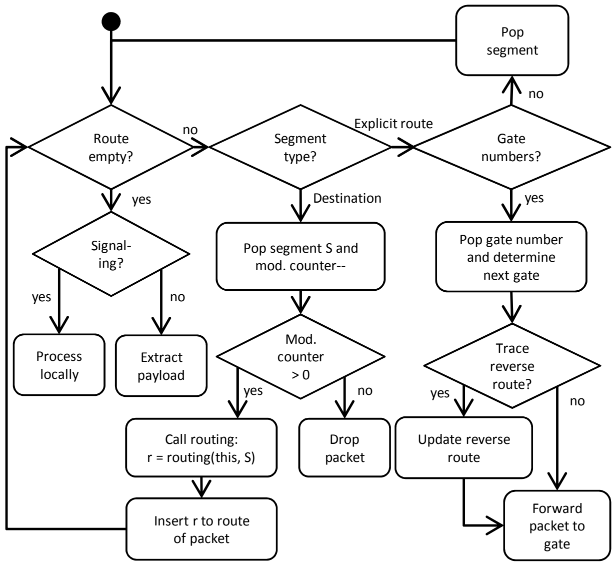


Figure 3: Forwarding node procedure without error cases

number, the packet is handed over to this gate for further processing. If the explicit route segment is empty, it is removed from the route and the procedure starts again. If the topmost segment is a destination segment, the forwarding node has to contact the routing component in order to get the next explicit route segment. Based on the received request data, the routing component calculates a route from the current forwarding node to the destination. The requirements given in the destination segment are used as requirements for the route. The resulting route from the routing component is added to the route of the packet by the forwarding component and the forwarding procedure is started again.

In order to reduce the impact of routing loops, a modification counter is decremented by one each time the route is changed in a way that loops might occur. As a result of this, a routing loop can be limited by forwarding only packets with a modification counter above 0 and dropping all other packets.

To allow a receiver of a packet to reply to the sender without knowing its exact address, the reverse route of a packet is recorded if the reverse route flag in the packet header is set. A forwarding node has to derive the reverse gate number from the gate chosen for the forward direction. The reverse route can be asymmetric to the forward route. Furthermore, an intermediate forwarding node, which is not able or allowed to record the reverse route, using explicit route segments, can add a destination segment to the reverse route instead. Such a reverse route can be used by the receiver of a packet to reply to the sender. The main benefit of using such reverse routes instead of an address is twofold. First, routing requests for reply packets are avoided and second, addresses for request-sending nodes are not required. The latter is useful for hosts acting only as clients. A server can reply by using the reverse route without forcing the client to have a unique and routable address. If a reply with a traced backward route is received by a client, it knows the route the received request packet has taken. In most cases, this route contains less destination segments and more explicit route segments. Therefore, the client can use this route for subsequent packets in order to reduce the routing overhead and the delay for its packets.

Section II.B describes that a destination segment of a route may define only an intermediate node of the route because not all network details (e.g., gate numbers) are known. Theoretically, multiple destination segments in one route are possible, which is comparable to loose source routing of IP. This is utilized in the use cases presented in the next sections.

Due to clarity reasons, Figure 3 does not show the error cases, which cause the drop of a packet. Example error cases are invalid formatted packet headers, invalid segment types, and data packets that routes end at forwarding nodes that do not represent a connection end point.

IV. USE CASES

Based on the FoG architecture and its network protocol, this section describes three different use cases showing the provisioning of QoS functions ranging from IntServ to DiffServ and a combination of both. For simplicity, the same example network is used in each case. Only the gate setup and the responsibility for the classification states (CS) differ.

Figures 4 and 5 show three networks with network 3 providing QoS functions in form of gates to networks 1 and 2. Gates are depicted as straight lines between the forwarding nodes (FN_i), which are shown as dots. The dotted lines represent connectivity through some other network, where the gate numbers are not known. Known gate numbers are labeled with small letters. Each network includes not only the forwarding component but also the routing component R_i as defined by the architecture. The routing components are depicted as extra round boxes with their network view inside.

The scenarios assume that at least one node within each network runs an instance of each component type. For simplicity reasons, each network can be seen as a network with just a single node running all three components. In real world deployments, edge nodes typically have to host instances of the forwarding component. There may be more instances of the forwarding component on core nodes, if a network uses FoG also as intra-network technology. The locations of the instances of the routing component differ depending on their implementations. An example may be a central routing instance within a network with proxies on edge nodes.

The following description further assumes that each component can be implemented and focuses on the architectural level. The protocols used by components, e.g., the routing component, depend on their specific implementation. More details about our implementation are given in Section VI. However, there may be other ways to implement the same architecture.

In the following, gate numbers are represented by italic letters. Routes and route segments are encapsulated with square brackets. For example, [[*b*]] is a route (outer brackets) with a single explicit route segment (inner brackets), which contains only the single gate number *b*.

A. IntServ gates

Figure 4 shows a scenario, in which networks 1 and 2 have requested QoS functions from network 3. Depending on the implementation, they might have used a simple request/response signaling procedure. Network 3 has set up one gate for each request with the gate numbers *b* and *c*. As stated in Section II.A, these numbers have to be unique only in the scope of FN_3. Thus, network 3 is allowed to select freely the numbers *b* and *c* according to the needs of the components implementation. Networks 1 and 2 have informed their routing component about these gates. For example, each gate represents a (virtual) link providing 100 MBit/s. The router, which is represented by FN_3, has to store the scheduling and signaling states required to provide and

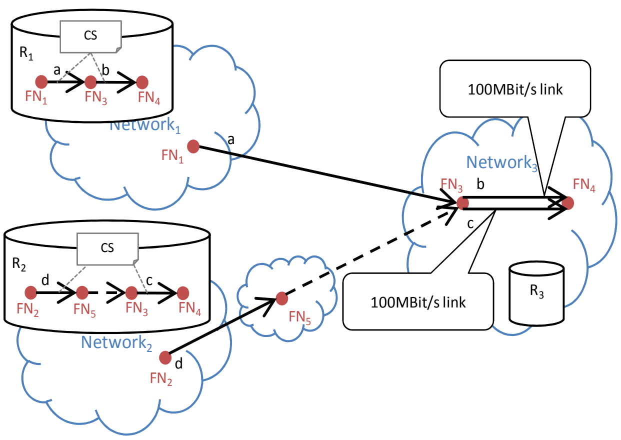


Figure 4: Gates representing IntServ reservations

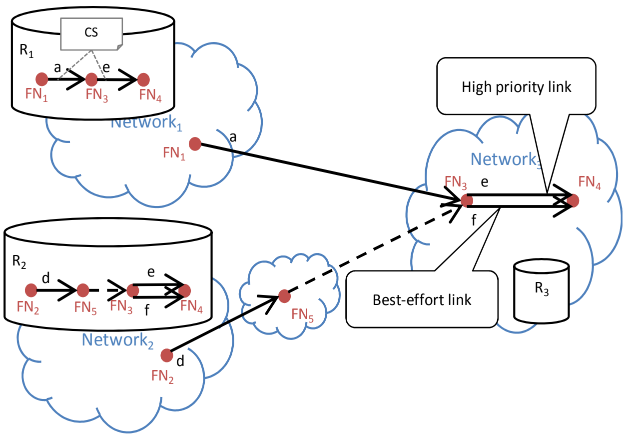


Figure 5: Gates representing DiffServ classes

enforce these QoS functions. However, the classification states are not stored by FN_3 but have been moved to networks 1 and 2, respectively. Their routing components know about gates *b* and *c*, and handle the decision about which flows are mapped to these gates.

If network 1 wants to establish a flow, the entity that is responsible for flow creation (e.g., a control node or a node at the network edge) starts sending a signaling message with a route, which contains only one destination segment containing the destination address and the requirements for the route, e.g., a minimum bandwidth of 10 MBit/s. In this example, the destination is FN_4. The packet with the route [[address(FN_4)]] is inserted into the forwarding component FN_1, which proceeds as described in Section III. Since the topmost segment of the route is a destination segment, it contacts the routing component R_1. In the given case, R_1 knows a route with all gate numbers to the destination. We assume that R_1 did not map too many flows on these gates and enough capacity is available. R_1 maps this new flow on the gates *a* and *b*, and updates its classification states. It returns the route [[*a*, *b*]] containing only one explicit route segment. FN_1 removes the destination segment from the packet and inserts this new route into the *route* field in the packet. FN_1 restarts the procedure with the explicit route segment as topmost segment. It pops the gate number *a* from the gate number stack and looks up this number in its list of outgoing gates. Afterwards, it hands over the packet to gate *a*. The gate transports the packet to the next hop via a link layer, e.g., Ethernet. The packet arrives at FN_3 with the route [[*b*]]. This forwarding node extracts (and removes) *b* from the stack and hands over the packet to gate *b*. FN_4 receives the packet with an empty route and processes the packet locally.

For network 2, the process is similar. However, the route required to reach FN_3 is different. R_2 calculates a route with three route segments: [[*d*], [address(FN_3)], [*c*]]. In contrast to the route calculated by network 1, one more request has to be used. FN_5 receives a packet with [address(FN_3)] as topmost segment, which triggers an additional route request at FN_5.

B. DiffServ gates

Figure 5 depicts a scenario, in which network 3 provides one gate with a high priority and a best effort gate with a low priority. The latter is included in the scenario to demonstrate the integration of non-QoS links in FoG. The routing components of networks 1 and 2 have slightly different views on this situation. R_1 only knows *e* and R_2 knows both *e* and *f*. The reason might be that R_1 decided to omit *f* or network 1 did not request a best-effort gate from network 3. Moreover, R_1 and R_2 have different strategies for tracking the flows mapped to these gates. R_1 stores the classification states as in the previous scenario. However, the criterion for using gate *e* differs. Instead of bandwidth metric, as in the previous example, R_1 might use a cost metric (e.g., money to pay to network 3) to decide which flow is important enough to justify the usage of gate *e*. R_2 does not limit the usage of gate *e* and *f* and does not store any classification states. It basically treats both as virtual best-effort links.

The calculated routes of this scenario are similar to the previous scenario. The main difference is the applied policy for selecting gates in R_1 and R_2. In addition to the previous scenario, R_2 shows the benefit of knowing a broader set of gates available for a link. Depending on the requirements for a route, R_2 can decide to use *e* or *f*. For example, it can return the route [[*d*], [address(FN3)], [*f*]] for flows without QoS requirements. Gate *e* can be used by R_2 without having to signal to FN_3. Furthermore, FN_3 does not have to know the details about flows and can just follow the gate numbers given in a packet. This reduces the load of the router hosting FN_3, *e*, and *f*.

C. Combined scenario

Both gate types can be combined in a single scenario. Such a scenario can be constructed by merging the two scenarios shown before. This combined scenario has four gates between FN_3 and FN_4 representing different QoS functions. In such a scenario, R_1 would have two options (since it does not know all gates) for a route from FN_3 to FN_4:

- Gate b: The usage is limited by the bandwidth, which is already reserved by R1 for other flows. By proper management of R1, a minimal bandwidth could be guaranteed.
- Gate e: The usage is limited by the costs, which network 1 is willing to pay for a flow (if it is charged by network 3). A certain amount of bandwidth cannot be guaranteed. However, the delay is minimized.

Which gate has to be chosen depends on the requirements for a flow and the requesting entity.

D. Discussion

Neither network 1 nor network 2 knows about the techniques used by network 3 to provide QoS. These implementation details are hidden from the routing, based on the abstract gate description, and from the forwarding, based on the used gate numbers.

In all scenarios, FN_3 does not store any classification states and does not know which flows are mapped to its gates by networks 1 and 2. It delegates the decision to these networks. However, the states required to enforce the characteristics of the gates remain in network 3. Thus, even though network 3 does not know, which and how many flows are mapped to a gate, it can enforce that the combined traffic does not use better QoS than requested. Another benefit of this delegation is a reduced signaling overhead. In particular, no signaling messages from networks 1 or 2 are required to inform FN_3 about new mappings.

The routes calculated by network 2 indicate an important case showing the difference between FoG and a combined MPLS / IP solution. The route [[*d*], [address(FN_3)]] could be implemented by the help of MPLS, handling the explicit route segment, and IP, handling the destination segment. However, a subsequent explicit route segment ([*c*] in the example) is not supported by MPLS and IP. In IP, the ingress router doing the IP forwarding has to have some classification states, which link a packet to the subsequent explicit route segment. However, FoG moves this state to other routers and thus reduces the number of states to be maintained by the ingress router.

The main use case of FoG consists of a network that provides some degree of QoS to its customers (its own end users and other networks). Thus, deploying FoG in order to implement a best effort network provides only limited advantages compared to IP. However, the degree of deployment is also critical for QoS scenarios. The delegation of states and decisions cannot be done only by network 3, since it requires the support of networks 1 and 2. Consequently, a partial deployment in today's Internet might not benefit from these two features. However, the more networks support FoG, the better is the exploitation of the advantages.

A migration strategy for introducing FoG to existing networks depends mainly on the legacy systems, which should be supported. For example, MPLS might be integrated by representing each "label switching path" (LSP) by gates.

V. Interoperability with current Networks

In the previous section, we discussed how FoG handles QoS functions. However, the architecture and its gates enable much more in-network functions. From an abstract point of view, gate numbers represent decisions of the routing, which are executed in the forwarding. Interpreting them as indices in a gate vector of a forwarding node is the simplest case. In general, gate numbers can be used to move information from an intermediate AS to the hosts at the border of the network or up-stream ASs without telling them about that shift. However, the usage of gate numbers is not limited to this.

A. Using layer 2 addressing

A gateway could store the MAC address of the destination node in a route. The forwarding would not need to create gates for all nodes, which are connected to an Ethernet domain. It would just use the MAC address given in the route. This enables stateless gateways, which do not have to store an address mapping (e.g., between IP and MAC addresses). The shift is transparent, since others see a list of gate numbers and do not need to know the meaning of each number. If the network has to prevent others from guessing MAC addresses, the routing can encrypt the address with a key, which is only known by the forwarding component. Based on this, the forwarding component can check whether a MAC address, given in a route, has actually been provided by the official routing component. In general, the resulting scheme, which is used to encode the routing decisions, can be adapted to the level of security, which is required for the network.

B. Network policies

Knowing the representation of decisions and how they are secured, introduces dependencies between the routing and forwarding component. In an inter-network scenario, this requires both components being operated by the same provider. Fortunately, the incremental routing process can be used to ensure this. By not announcing gates, others are forced to involve the forwarding of an AS in the route calculation, as shown in use case 1. Thus, ASs have the opportunity to insert self-generated route segments into the route. How these route segments encode the decisions for the forwarding instance is solely up to the AS

C. Protocol tunnels

Since gates hide the implementation of the QoS-aware data transport, any technology (e.g., MPLS) can be used. Therefore, gates can be interpreted as tunnels transporting packets through networks. By constructing explicit routes, tunnels are concatenated. Such a tunnel semantic can be implemented by having "tunnel gates". In particular, such gates are useful for hiding intermediate structures and to shorten route descriptions in packets. For example, this approach can be utilized in order to implement interoperation with IP based networks.

Figure 6 shows an example scenario consisting of two FoG networks, which are connected via an IP based transit network. If node A of network FoG_1 wants to send packets to node D, located in network FoG_2, it can use the route description [[*1, 5, 4*]]. This directs packets along gate 1, 5 and 4. Gate 5 is located on FN_1 and implements the desired tunneling function by splitting each FoG packet into payload fragments, which are headed by an additional fragmentation header. These fragments have to fit into IP packets and are sent via the IP based network interface of FN_1 to the IP based network interface of FN_2. For this purpose, gate 5 has to know the IP address of FN_2. FN_2 detects such tunneling packets by checking the "protocol" field of the IP header from received packets for a FoG specific ID. As a result of this, FN_2 is able to reconstruct the original FoG packets and continues their processing as described in Section III in order to forward them towards their destination in the FoG network.

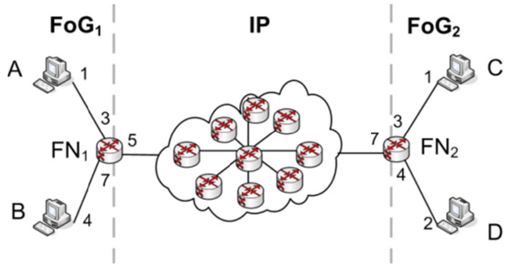


Figure 6: Two FoG networks are connected via an IP transit network

D. Gateway interoperability with IP

The goal of gateway interoperability is to provide a transparent mechanism for direct data exchange between a FoG and an IP based network. Figure 7 shows an example where applications on node A and B exchange directly data via FN_2 with applications on the IP based node C.

For addressing purposes on the IP side, both layer 3 addresses and layer 4 port numbers are used. Therefore, such a data exchange has to use TCP, UDP or SCTP (or any other transport protocol, which uses port numbers) on IP side.

If an *IP client* wants to send data to a *FoG service*, a dedicated gateway application has to be started before on the gateway node (FN_2 in Figure 7). The application owns a network server socket towards the IP network and receives IP based packets from IP clients. The application transforms these packets into FoG packets and sends them towards the service providing node in the FoG network. For this purpose, the application has to know a static mapping from its network socket (providing the service end point towards the IP network) to a FoG service name. For each new IP based client, the application stores an additional mapping between a dedicated FoG connection and IP status data, which allows a reverse mapping for answer packets from the FoG service.

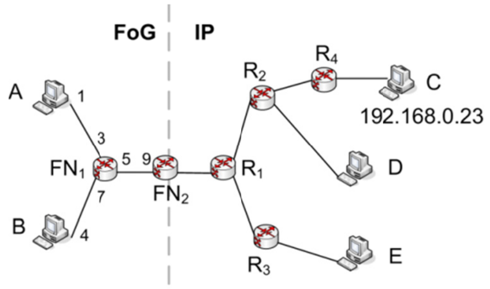


Figure 7: Direct connection between a FoG and an IP based network

For a transmission from a *FoG client* to an *IP based service*, a dedicated gateway application is used. Similar to the previous paragraph, it is located on FN_2. Each time a FoG client starts a communication to an IP based service, such an application instance is created. The FoG client sends its packets towards the gateway node. For addressing the correct receiver at IP side, the route in each packet includes a special destination route segment, which describes the IP receiver. For each FoG connection, the gateway application has to store a mapping from a FoG client to an IP based destination. This is also used for deriving the receiver at FoG side for each answer packet from an IP service.

VI. Evaluation

The applicability of FoG to combine various types of QoS for our use cases was shown in the previous sections. In the following, we present simulation results analyzing the scalability of FoG for large networks. The analysis focuses on one type of gates such as prioritized relaying. Thus, the results are general and can be interpreted easily for situations with more gate types by combining individual analyses for each gate type.

We measured three important metrics:

- Length of explicit routes: In order to evaluate the overhead induced by the FoG packet header, we measured the length of explicit end-to-end routes. This overhead is compared to the overhead of IP.
- Size of the routing graph required by a FoG node to store information about networks: As shown by the previous use cases, the size of the routing graph stored by the routing component depends on the number of gates known from others. We measured the graph size for different setups and compare them to the situation for source routing and IP networks.
- Number of request for (partial) routes: Since the knowledge of the routing components can vary, the incremental routing process may have to call the routing component more or less often. We measured the number of requests in order to approximate the routing overhead.

For the functional evaluation and measurements, we have implemented the entire FoG architecture in a Java-based discrete event simulator called "FoGSiEm". It can be switched to an emulator mode that handles events in real time. The FoG emulator includes interoperability solutions combining FoG and IP based networks [16]. The software supports Windows, Linux and OS X and is available as open-source [17]. It includes the gateway interoperability as described in Section V.D with client and server sockets on FoG and IP side. We demonstrated this feature for web browsing and video streaming [18].

A. Simulation setup

The evaluation relies on simulations of large inter-networks. We derived the simulated networks from generated graphs modeling the Internet on the level of autonomous systems.

1) Network graph

The network used for evaluation matches the characteristics of today's Internet but has a smaller size in order to enable simulations. Its network graph has been generated with the GLP algorithm implemented in BRITE [19] and the parameters derived in [20]. Therefore, the graph has similar characteristics as the real world Internet graph on the autonomous system level. It consists of 5,000 nodes and 12,437 links between them. In addition, a different graph generated with the default parameters (α=0.45, β=0.64) of BRITE (5,000 nodes and 8,974 links) was used for simulation. Since the results do not differ significantly, only the results from the first graph are presented.

Each node of a graph represents an autonomous system. In the simulation, such a node is represented by one forwarding node per edge and one forwarding node representing the core of an autonomous system. The edge forwarding nodes are linked to the core forwarding node with a star topology, which represents the internal connectivity within the autonomous system. Each link between autonomous systems is represented with a link between two edge forwarding nodes.

2) Announcing gates to neighbors

The use cases discussed in the previous sections comprises only a small excerpt from a whole network. For the evaluation of the routing graph sizes, we need to define how many gates are known to the routing components of each FoG node of a large network. In the following, the algorithm used to configure the routing component instances used by our simulation is described.

We assume that FoG nodes join groups. Those group members trust each other and exchange information about gates with them. More specific, all members of a group announce their gates and gate numbers to all other members of the group. They are allowed to reuse these gates in the incremental routing process of Section II.B.

Groups are defined by clustering the nodes of a network. Clusters are generated by choosing cluster heads randomly at the start of a simulation. Nodes that are no cluster head join the cluster represented by the nearest cluster head. If two or more cluster heads are available within the same distance, one is chosen randomly. If no node decides to become a cluster head, a single node is selected randomly by the simulation.

Within a cluster, routing components exchange their knowledge. With nodes outside of a cluster, only abstract information is exchanged. This abstract information enables nodes to determine the shortest path to a destination but does not include any gate numbers. Thus, this information is comparable to the information exchanged in IP networks, e.g., between BGP entities.

The clustering is a possible and simple way to define the announcement policy. In reality, nodes will follow a more complex scheme based on the business plans of their operators. Our approach reduces the complexity but includes typical setups of today's networks as shown later on.

B. Overhead due to explicit route

The route length of explicit route segments is a specific concern. The more hops a packet has to travel, the more gate numbers are required and the longer is the header. In order to estimate the FoG protocol overhead, the length of explicit routes for large-scale scenarios has been analyzed.

1) Results

The analysis is based on the lengths of explicit routes of 100,000 connections between randomly chosen FoG nodes. Figure 8 shows the empirical distribution function of the route lengths. Since each intermediate node uses three gate numbers and each end node uses two gate numbers in order to encode its routing decision, only specific route lengths such as 4 and 7 are possible. An end-to-end route contains 11.95 gate numbers in average. The average number of hops between two FoG nodes $L = 3.65$ matches the expectations for an Internet-like graph [21].

2) Discussion

Figure 8 shows the distribution of the route lengths in number of gate numbers. For a comparison with the address length of IP, an encoding of the routes is required. If a gate number, the route length field, the route segment length field, and the route segment type field from a FoG header (cp. packet format in Section III) are encoded with a single octet, a route with 13 gate numbers requires 16 octets in total. Thus, such a route would require the same number of octets

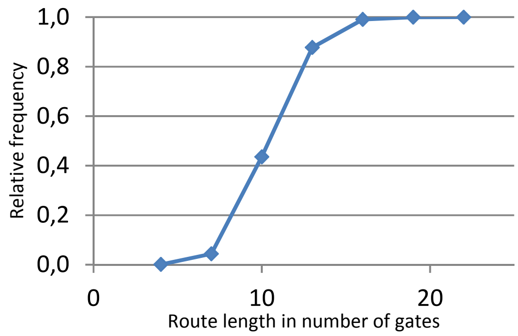


Figure 8: Empirical distribution function of end-to-end route lengths

as an IPv6 address. According to Figure 8, 87% of the routes have 13 gate numbers or less. Thus, most routes have a length, which is shorter or equal to an IPv6 address.

Since the setup requires three gate numbers to cross one node, the encoding with one octet per gate number seems to be realistic (in other words: three octets per hop). Thus, the overhead induced by transporting explicit routes seems to be acceptable in comparison with IPv6.

In average, only half of the gate numbers are transported over links, because gate numbers are successively removed from the packet header by the forwarding component. Since FoG may trace a reverse route, which increases the packet size again, the average route length mentioned above includes both routes. Thus, two routes with a total average size of 12 gate numbers replace the source and destination addresses of an IP packet. Even if the encoded size of a gate number is doubled to two bytes, 87% of the FoG packets contain less overhead than an IPv6 packet.

C. Routing graph size

In order to evaluate the scalability of the routing component, we simulated different setups of FoG's routing by varying the knowledge base of the routing components. Therefore, we cluster a network to model groups of trusted nodes as described in Section V.A.2. There are two extreme cases of the clustering:

- There is just a single cluster including all nodes. All network elements know the entire network graph and, thus, can use all gates for constructing routes. Such nodes can calculate complete end-to-end routes. This is equivalent to source routing.
- Each node forms its own cluster. All nodes know only their own gates. Additionally, they know the other network parts in an abstract way in order to find a shortest path. This setup is comparable to the operations of IP, since routing is done in a hop-by-hop manner.

In our simulations, a node knows the gates and forwarding nodes of the cluster it belong to and can calculate only partial routes for the cluster. For the construction of an end-to-end route, FoG has to combine multiple partial routes as described by the incremental routing process presented in Section II.B. Basically, an ingress node of a cluster calculates the route through its cluster, since it has the required knowledge.

1) Results

The results shown in the following are average results derived from 10 simulation runs with randomly chosen cluster heads. Error bars indicate the min/max results of the 10 runs. Each run contains 25,000 flows, which exist between randomly chosen nodes.

Figure 9 shows the number of gates and forwarding nodes (edges and vertices) of the detailed routing graph a node has to store. Figure 10 shows the same numbers for the abstract graph, which does not include gate numbers. For example, R_2 in Figure 4 has 4 vertices and 2 edges in its detailed graph. In its abstract graph, R_2 has one gate (dotted one) and its start and end forwarding nodes. Since a node has to store both graphs, Figure 11 shows the average graph sizes (vertices plus edges) of both graphs and the sum of both.

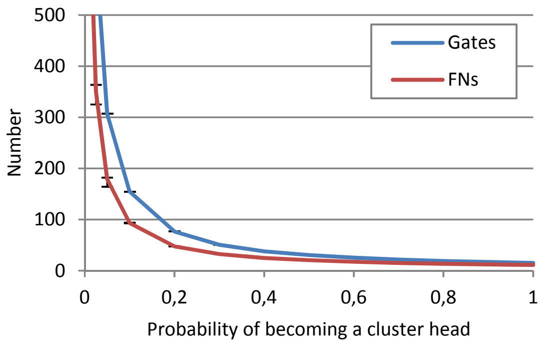


Figure 9: Number of gates and forwarding nodes (FNs) for detailed graph. Maximum values for probability zero: 76k gates and 30k FNs

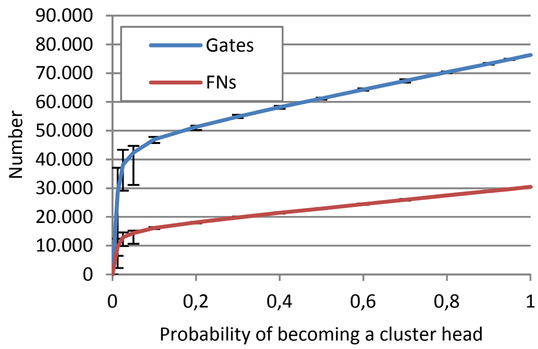


Figure 10: Number of gates and forwarding nodes (FNs) for abstract network graph.

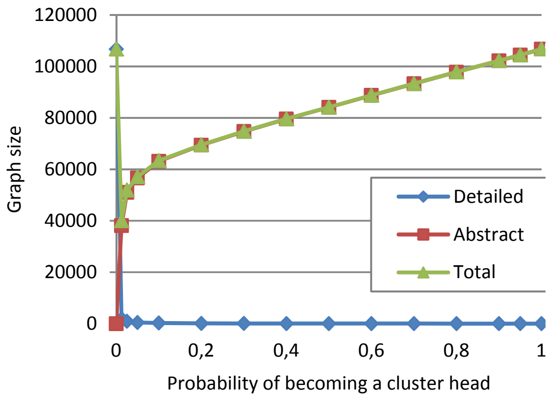


Figure 11: Average routing graph sizes

The x-axes of these three figures show the probability of a node to become a cluster head at the start of a simulation. In the source-routing setup (probability is zero) on the left hand side, each routing component knows the entire network with all gates. As expected, the graph for such approaches is very big. For hop-by-hop setups (probability is one), the amount of detailed information is significantly smaller as shown on the right hand side. Each node knows only the gates it hosts and remote gate numbers are not known. However, the abstract graph is rather big, since a lot of clusters and gates connecting them are known. In total, a routing component in a network configured in a hop-by-hop manner has to store slightly more information than a routing component in a source-routing network.

2) Discussion

The evaluation shows that there are a lot of configurations beside the extreme cases: source- and hop-by-hop routing. They require smaller overall routing component graphs. For example, the average routing component size for a probability of 0.025 with approximately 52k elements is half of the average sizes of the two extreme cases. From the perspective of the use cases, these intermediate configurations seem to be the more realistic ones. Some nodes request gates from remote peers and their routing components have the knowledge about these gates. They can combine them to routes for their data flows. These nodes are not interested in all gates as in the source routing configuration. They delegate decisions about route parts, which are “unimportant” for them, to others by using the incremental routing process. For the important parts, they exploit FoG’s partial routes in order to define these route parts explicitly.

In relation to BGP, the results represent a worst-case scenario, because the nodes store information about all links. A BGP peer would filter the information according to its policy (e.g., only shortest path) and not announce all information to its peers.

D. Number of routing requests

The overall overhead of the routing is not only determined by the graph size. It also depends on the number of route calculations within this graph. Thus, we measured the average number of requests required to setup one end-to-end route. Figure 12 shows the results that are based on 10 simulation runs and 25,000 flows per run.

1) Results

The x-axis of Figure 12 shows the probability that a single node decides to become a cluster head at the start of a simulation run as described in the previous section. If the probability is zero, the source-routing configuration induces a single route calculation per flow. Between zero and 0.1, the number of route requests increases disproportionately. Above 0.1, the number continues to increase nearly linearly. If the probability is one, the hop-by-hop configuration requires 4.7 requests on average. This number includes one request per hop (3.7 hops on average as mentioned in Section V.B) plus a last calculation at the destination in order to determine the forwarding node within the destination node.

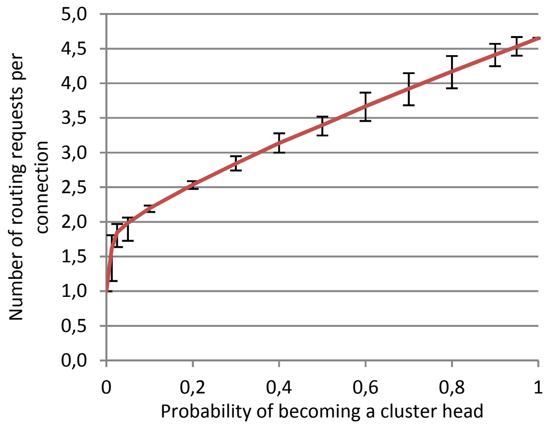


Figure 12: Average number of requests for (partial) routes in order to construct one end-to-end route.

2) Discussion

A request has to be answered by a routing component with a partial route towards the destination. This answer has to be calculated by an algorithm based on the graphs analyzed in the previous section. Thus, the computational overhead depends on the number of requests, the graph sizes and the applied algorithms. If the algorithm is fixed and its runtime complexity depends on the graph size, a trade-off between the number of requests and the graph size is required. In our example, we used the Dijkstra for each calculation. Thus, a probability of 0.0125 provides an optimal trade-off with a graph size of around 40k and 1.6 requests.

However, such a trade-off depends on the algorithm. Thus, different algorithms may lead to different optimal trade-offs. For example, if a forwarding information base (FIB) is introduced, the runtime of the algorithm does not depend directly on the graph size anymore. We refrain from analyzing FIBs in this article, because they depend strongly on the distribution of addresses. Since FoG supports a variety of implementations of the routing component, neither the algorithm nor the addressing or the type of addresses are fixed. For example, FoG can operate with IP addresses, BGP or with HRM with its hierarchical addresses. Thus, the optimal trade-off depends on the implementation. Our analysis underpins the flexibility of FoG and the new opportunities for designing routing components.

VII. RELATED WORK

The question about how to provide an adequate QoS for networks has a long research history. A survey about today’s approaches is given in [22]. As discussed in the introduction, IntServ [3] and DiffServ [6] can be used to provide QoS. However, they do not support the transparent movement of QoS classification states among nodes as it is enabled by the FoG architecture. By combining MPLS with IP, many QoS use cases can be implemented. However, Section IV.D points out important cases where the combination of IP and MPLS does not allow the movement of classification states. Furthermore, the FoG architecture does not require a standardization of gate numbers as it is required for the IP TOS field in an inter-network scenario [23].

In general, FoG implements a strict split of packet processing into forwarding and routing components similar to PFRI [10], which uses labeled "channels" and anonymous nodes to model the available connectivity of lower layers. The channel labels are globally bijective and are comparable to FoG gates representing lower layer connections. However, the PFRI channel labels are not comparable to gate numbers, since gate numbers in FoG are only bijective in the scope of a forwarding node.

Other forwarding approaches use a stack of locally valid numbers to describe routes, e.g., PARIS [24], Sirpent [25] or Pathlet [11]. In PFRI [10], these numbers are even globally unique in order to enable the end host to specify a loose source route based on links. Such entries in a forwarding table represent either virtual [11] or physical [26] next hops. In a FoG network, each gate has a locally unique number in the scope of a node. The number assignment is controlled by the corresponding parent node. Globally unique numbers are not mandatory in a FoG network. But server application must have globally unique addresses to be addressable by clients, which can be located everywhere in the entire network. Furthermore, FoG uses names for forwarding nodes and not gates, because we expect a network to have more links than nodes.

Some older architecture approaches [24, 25] are focused on intra-networks. Other proposals for new inter-network architectures focus more on the overall architecture and do not address scalability of state distribution, e.g., NewArch [27], IPC [12], RNA [28]. The Pathlet approach, as the newest one, deals specifically with policy issues in inter-network routing. However, QoS and other application requirement aspects are not discussed in detail. In the Pathlet descriptions, QoS is only mentioned but any details about how to integrate IntServ or DiffServ, and a network protocol are missing. The FoG architecture includes these aspects with respect to scalability. It is focused mainly on inter-networks. QoS path reservation protocols, e.g., RSVP or NSIS [29], signal QoS requirements. One of these protocols or similar approaches are suitable to trigger the setup of gates.

The combination of function blocks can either be done at design time (similar to Netlets [30]) or at runtime (similar to SONATE [31]). The architecture of FoG decides everything during runtime. It reacts dynamically on inputs from applications (e.g., QoS requirements) and network states (network policies and available routes). While SONATE [31] is focused on selecting and composing function blocks on end hosts, FoG can also be used to integrate function blocks residing on intermediate routers.

The impact of using heap structures for header information as proposed by the Role-based Architecture [32] has not been analyzed. The "roles" of this architecture are comparable to gates. However, RoleIDs (and, thus, role addresses) are not comparable to gate numbers because a RoleID contains an identifier for the type of function and not for a special instance.

VIII. Conclusion and Outlook

In this article, we presented the Future Internet architecture "Forwarding on Gates" (FoG). It applies its own network protocol in order to provide three different ways for selecting a route for packets: define the route explicitly, define the route indirectly based on the destination address plus requirements or – as third way – use a combination of both. This enables the movement of classification states between routers. IntServ and DiffServ are merged by introducing QoS functions, which are represented by directed gates in the FoG architecture. Routes can be defined by using the gates without knowing about their implementation. FoG enables the flexibility to move classification states from the router, which implements a QoS function, to other routers, which take over the mapping of flows to QoS functions. This delegation of mapping decisions reduces the amount of required signaling messages.

Based on three use cases, we described the setup of gates in IntServ, DiffServ and mixed scenarios. Although the route length is dynamic, the protocol overhead remains low. A protocol simulation in a large-scale network with 5000 nodes showed that 87% of the routes are shorter than an IPv6 address. Simulations of a basic version of the routing component showed the flexibility of FoG by using partial routes. Depending on the knowledg a node and its routing component have, FoG can be used for a trade-off between hop-by-hop routing (with many routing requests) and source routing (with only one routing request). FoG's routing component implementation can exploit this flexibility and can use new address types or routing algorithms.

The results presented in this article show the flexibility of FoG in providing QoS in a scalable way. It indicates that the FoG architecture is a promising basis for a network layer architecture that can replace IP in a Future Internet. Since a complete replacement of IP seems to be unrealistic, Section V outlined several interoperability solutions for partial deployment scenarios. They are similar to the interoperability solutions for IPv4 and IPv6. Thus, running FoG in parallel to IP is an deployment option as well.

In the future, we plan to use route repair techniques known from MPLS to evaluate the robustness of FoG routes against link and node failures. Additionally, we will evaluate the FoG architecture in comparison to other existing architectures for selected Internet-like scenarios. Furthermore, we plan to extend the existing qualitative evaluation [15] of the routing component infrastructure "Hierarchical Routing Management" [14] by measuring its quantitative advantages and management overhead for selected complex network scenarios.

ACKNOWLEDGMENT

This work is funded by the German Federal Ministry of Education and Research under the project G-Lab_FoG (code 01BK0935). The project is part of the German Lab [33] research initiative.

REFERENCES

[1] F. Liers, T. Volkert, and A. Mitschele-Thiel, "The Forwarding on Gates architecture: Merging IntServ and DivServ," Proc. of International Conference on Advances in Future Internet (AFIN) 2012, pp. 7-13, August 2012.

[2] Cisco Systems, "Cisco Visual Networking Index: Forecast and Methodology, 2010–2015," white paper, 2011, http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/white_paper_c11-481360.pdf, last access: December 2013.

[3] R. Braden, D. Clark, and S. Shenker, "Integrated Services in the Internet Architecture: an Overview," IETF, RFC 1633, June 1994.

[4] C. Deleuze and Serge Fdida, "A scalable IntServ architecture through RSVP aggregation," Networking and Information Systems Journal, vol. 2, no. 5-6, pp. 665-681, 1999.

[5] B. E. Carpenter and K. Nichols, "Differentiated service in the Internet," Proc. IEEE, vol. 90, no. 9, pp.1479-1494 , 2002.

[6] S. Blake et al., "An Architecture for Differentiated Services," IETF RFC 2475, December 1998.

[7] Y. Bernet et al., "A Framework for Integrated Services Operation over DiffServ Networks," IETF RFC 2998, November 2000.

[8] X. Masip-Bruin et al., "The EuQoS System: A solution for QoS Routing in Heterogeneous Networks," IEEE Communications Magazine, vol. 45, no. 2, pp. 96-103, February 2007.

[9] F. Liers, et al., "GAPI: A G-Lab Application-to-Network Interface," 11th Würzburg Workshop on IP: Joint ITG and Euro-NF Workshop "Visions of Future Generation Networks" (EuroView2011) , Würzburg, Germany, August 2011.

[10] K. L. Calvert, J. Griffioen, and L. Poutievski, "Separating Routing and Forwarding: A Clean-Slate Network Layer Design," Proc. of the Broadnets 2007 Conference, pp. 261-270, September 2007.

[11] P. B. Godfey, I. Ganichev, S. Shenker, and I. Stoica, "Pathlet Routing," Proc. of SIGCOMM 2009, pp. 111-122, August 2009.

[12] J. Day, I. Matta, and K. Mattar, "Networking is IPC: A Guiding Principle to a Better Internet," Proc. of ReArch'08, Article no. 67, Madrid, Spain, December 2008.

[13] A. Reitzel, "Deprecation of Source Routing Options in IPv4, IETF," Internet-Draft, August 29, 2007.

[14] T. Volkert, M. Osdoba, A. Mitschele-Thiel, and M. Becke, „Multipath Video Streaming Based on Hierarchical Routing Management," Proc. of 27th IEEE International Conference on Advanced Information Networking and Applications (AINA), Barcelona/Spain, March 2013.

[15] T. Volkert and A. Mitschele-Thiel, "Hierarchical routing management for improving multimedia transmissions and QoE," Proc. of IEEE International Symposium on a World of Wireless Mobile and Multimedia Networks (WoWMoM), San Francisco/USA, June 2012.

[16] F. Liers, T. Volkert, and A. Mitschele-Thiel, "Scalable Network Support for Application Requirements with Forwarding on Gates," Demo at 11th Würzburg Workshop on IP: Joint ITG and Euro-NF Workshop "Visions of Future Generation Networks" (EuroView2011), Würzburg, Germany, August 2011.

[17] "FoGSiEm" on GitHub, web site: https://github.com/ICS-TU-Ilmenau/fog/wiki, last access: December 2013.

[18] T. Volkert and F. Liers, "Video transcoding and rerouting in Forwarding on Gates networks," 12th Würzburg Workshop on IP: ITG Workshop "Visions of Future Generation Networks" (EuroView2012), Würzburg, Germany, August 2012.

[19] A. Medina, A. Lakhina, I. Matta, and J. Byers, "BRITE: an approach to universal topology generation," In IEEE MASCOTS, pp. 346–353, Cincinnati, OH, USA, August 2001.

[20] H. Haddadi, D. Fay, S. Uhlig, A.W. Moore, R. Mortier, A. Jamakovic, and M. Rio, "Tuning Topology Generators Using Spectral Distributions," Proc. of SIPEW, pp. 154-173, 2008.

[21] CAIDA, "Comparative analysis of the Internet AS-level topologies extracted from different data sources," http://www.caida.org/~dima/pub/as-topo-comparisons.pdf, last access: December 2013.

[22] D. Vali, S. Paskalis, L. Merakos, and A. Kaloxylos, "A Survey of Internet QoS Signaling," IEEE Communications Surveys & Tutorials, vol. 6, Fourth Quarter, pp. 32-43, 2004.

[23] Cisco Systems, "Implementing Quality of Service Policies with DSCP," http://www.cisco.com/application/pdf/paws/10103/dscpvalues.pdf, last access: December 2013.

[24] I. Cidon, and I. S. Gopal, "PARIS: An approach to integrated high-speed private networks," International Journal of Digital ans Analog Cable Systems, pp. 77-85, 1988.

[25] D. R. Cheriton, "Sirpent: a high-performance internetworking approach," Proc. of ACM SIGCOMM '89: Symposium Proceedings on Communications architectures & protocols, pp. 158-169, 1989.

[26] H. T. Kaur, S. Kalyanaraman, A. Weiss, S. Kanwar, and A. Gandhi, "BANANAS: An evolutionary framework for explicit and multipath routing in the Internet," Proc. ACM SIGCOMM 2003, pp. 277-288, FDNA Workshop, August 2003.

[27] D. Clark, K. Sollins, J. Wrolawski, D. Katabi, J. Kulik, X. Yang, R. Braden, T. Faber, A. Falk, V. Pingali, M. Handley, and N. Chiappa, "NewArch: Future Generation Internet Architecture," Technical Report, 2003, http://www.isi.edu/newarch/iDOCS/final.finalreport.pdf, last access: December 2013.

[28] J. Touch, and V. Pingali, "The RNA Metaprotocol," Proc. IEEE International Conf. on Computer Comm. (ICCCN), pp. 1-6, August 2008.

[29] R. Hancock et al., "Next Steps in Signaling (NSIS): Framework," IETF, RFC4080, June 2005.

[30] L. Völker, D. Martin, C. Werle, M. Zitterbart, and I. El Khayat, "Selecting concurrent network architectures at runtime," Proc. of the 2009 IEEE international conference on Communications, ser. ICC'09. Piscataway, NJ, USA: IEEE Press, 2009, pp. 2124–2128.

[31] P. Müller and B. Reuther, "Future internet architecture - a service oriented approach," it - Information Technology, vol. 50, no. 6, pp. 383–389, 2008.

[32] R. Braden, T. Faber, and M. Handley, "From protocol stack to protocol heap: role-based architecture," SIGCOMM Comput. Commun. Rev., vol. 33, no. 1, pp. 17-22, January 2003.

[33] German Lab, web site: http://www.german-lab.de, last access: December 2013.

Intelligent Learning Techniques applied to Quality Level in Voice over IP Communications

Demostenes Zegarra Rodriguez, Renata Lopes Rosa, and Graça Bressan
Department of Computer Science and Digital Systems
University of São Paulo, SP - Brazil
Email: demostenes@larc.usp.br, rrosa@usp.br, gbressan@larc.usp.br

***Abstract*—This paper presents a method for determining the quality of a Voice over IP communication using machine learning techniques. The solution proposed uses historical values of network parameters and communication quality in order to train the different learning algorithms. After that, these algorithms are able to find the quality of the Voice over IP communication based on network parameters of a specific period of time. Intelligent and other machine learning algorithms take as input a baseline file that contains some values of network parameters and voice coding, associating an index quality for each scenario according to ITU-T Recommendation G.107. The tests were performed in an emulated network environment, totally isolated and controlled with real traffic of voice and realistic IP network parameters. The quality ratings obtained for the learning algorithms in all the scenarios were corroborated with the results of the algorithm of ITU-T Recommendation P.862. The results show the reliability of the four learning algorithms used on the tests: Decision Trees (J.48), Neural Networks (Multilayer Perceptron), Sequential Minimal Optimization (SMO) and Bayesian Networks (Naive). The highest value of reliability for determining the quality of the Voice over IP communications was 0.98 with the use of the Decision Trees Algorithm. These results demonstrate the validity of the method proposed.**

Keywords-QoS; VoIP; Machine Learning; MOS; E-Model; PESQ.

I. INTRODUCTION

The quality of a Voice over IP (VoIP) communication does not have the quality levels of the conventional circuit-switched telephony; thus, users who do not have an acceptable user experience with VoIP calls continue using traditional telephony. For this reason, the study of methods for evaluating quality of a VoIP communication is very important because it allows network resources to be reallocated to improve communication quality [1].

Initially, the determination of the quality of a VoIP communication was conducted by subjective tests, resulting in a quality score called MOS (Mean Opinion Score); ITU-T Recommendation P.800 [2] describes the requirements and methodology followed in these tests. Later, some objective methods were employed, such as ITU-T Recommendation P.862 [3] or PESQ (Perceptual Evaluation of Speech Quality), which determine an index named MOS-LQO (MOS-Listening Quality Objective), which is the result of the comparison of the original speech or reference signal and the degraded speech signal. Also, nonintrusive methods were developed, such as ITU-T P.563 [4], the algorithms of which do not need a voice signal reference.

Other metrics of voice quality [5][7] do not consider the voice signal, for determining the quality index models based on network parameters such as the E-Model [5] are employed, used in network planning and to configure the rate of VoIP communications.

Machine Learning algorithms have been used to determine the quality of multimedia services, thus trying to ensure a better quality of service [8][9]. For evaluating voice quality in VoIP services, [10][13] show how neural networks are used for monitoring this service, but other learning techniques are not studied and do not detail how the training file used was built. It is worth noting that, the E-Model is not sufficient to predict the voice Quality Level, because sometimes a parameter is missing and it is not possible to measure the QoS with the E-Model. Conversely, with machine learning, if one parameter is missing, it is possible to measure the QoS. In this context, this paper uses as a training file built based on ITU-T Recommendation G.107, better known as E-Model, considering some network scenarios that were extracted from real traffic. As a consequence, the results attained high levels of reliability.

The algorithms used in this study come from different approaches in the artificial intelligence sub-area devoted to the study of machine learning to predict the quality level of a service. These algorithms are: Decision Tree (J48 - C4.5 algorithm), Bayesian networks (Naive Bayes), Sequential Minimal Optimization (SMO) and Neural Networks (Multilayer Perceptron), and are used to determine the quality of a VoIP communication in a sample interval of 8 seconds. The reliability of the algorithms specified for this application was measured. Network training was performed using a 650-case file, prepared by the E-Model algorithm. Each line of the file contains the network parameters: transmission rate, delay and packet loss probability, and the value of voice quality index (MOS), which is the result of the E-Model algorithm.

This work considers the encoding rates of 64 kbps for ITUT G.711 [14] codec and rate of 8 kbps for ITU-T G.729 [15], respectively, and also considers the intrinsic values that these codecs have in the scenario with packet loss. The tests were performed in a scenario of IP network emulation, where a VoIP communication is established and different network parameters are programmed, in order to study the quality degradation of voice communication for each test scenario. For network emulation purposes, a network emulator software was used. Thus, the parameters of packet loss probability and delay in an IP network were changed.

This article is divided as follows: Section II makes a theoretical revision of the machine learning algorithms used in this work. Section III deals with the voice quality assessment methodologies. Section IV presents the test scenario, the methodology followed for the tests and the parameters evaluated. Section V shows the experimental results and discussions, and finally, Section VI presents the conclusions and future work.

II. Algorithms used in the Determination of Voice Communication Quality

Artificial intelligence and machine learning appeared in the mid 1950s. The ability to learn is the main characteristic of artificial intelligence. This section presents the different algorithms used for training the IP network to determine the quality of VoIP communications.

The proposal of this work is to study how the artificial intelligence can help to find the adequate level of quality for a voice communication, because in [16] the artificial intelligence is used to find key metric for QoS in VoIP applications, but studies only the packet loss. In [17] the packet loss and jitter are considered, but only some artificial algorithms are studied.

A. Decision Tree Classification

Decision Trees [18] are tools that can be used for giving the agent the ability for both learning and making decisions. The decision tree takes as input a situation described by a set of attributes and returns a decision, which is predicted by the value of the input attribute. The input can be both discrete or continuous values. Only discrete values are used herein. The learning of discrete values is called classification.

To better understand the operation of a decision tree, it is considered the problem of choosing the correct QoS (Quality of Service) regarding to an IP network parameter, for instance, the delay, as shown in Figure 1.

B. Bayesian Classification

The Bayesian classification [19] algorithm has its name because it is based on Bayes Theorem probability. It is also known as Naive Bayes classifier or only Bayes algorithm. The algorithm aims to calculate the probability of an unknown sample belonging to each of the known classes. This type of prediction is called statistical classification; it is completely based on probabilities.

A feature of this algorithm is that it requires a data set previously classified. Based on this preliminary data set, also called training set, the algorithm takes as input a new unknown, i.e., which has no classification, and returns as output the most likely class for this sample according to probabilistic calculations.

The probability model for a classifier is a conditional model over a dependent class variable C with a small number of classes, dependent on several feature variables F_1 through F_n.

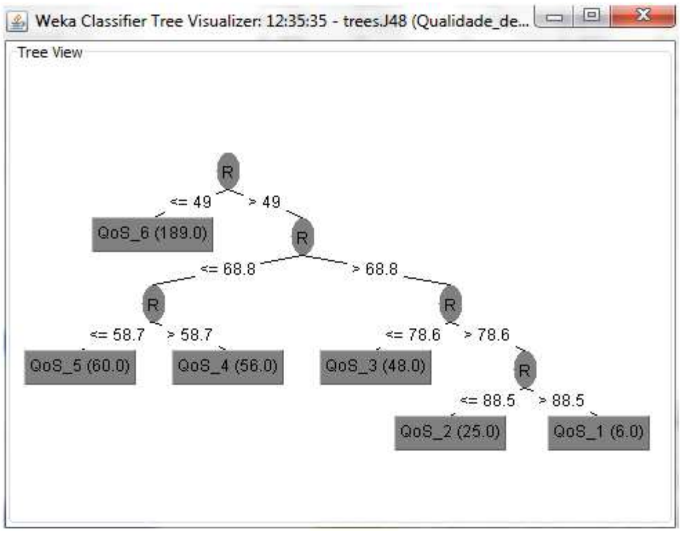


Figure 1: Tree algorithm classification used to determine the QoS, regarding the delay parameter.

C. Multilayer Perceptron

Rosenblatt [20] introduced the perceptron as a simple algorithm for neural network, capable of linearly classifying separable patterns. The operation of a perceptron (artificial neuron) shows that:

- The neuron is responsible for calculating the combination of inputs and weights, and then applies an activation function that determines the effective output of the neuron.
- The training is performed through the presentation of known inputs and outputs (supervised learning) and through adjusting the weights with specific algorithms.

Multilayer Perceptron Networks (MLP) is computationally more powerful than networks without hidden layers. MLP can handle data that are not linearly separable.

The processing performed by each neuron is defined by the combination of the processing performed by the previous neurons layer connected to it. From the first hidden layer to the output layer, implemented functions become increasingly complex. These functions define how the space-making division is made. There are several algorithms to train MLPs.

Among these, the most popular learning algorithm for training these networks is back propagation [21]. This is a supervised algorithm that uses the desired output for each input provided to adjust the parameters, called weights of the network. In addition, the adjustment weights use the backpropagation gradient method to define the corrections to be applied.

Figure 2 illustrates a perceptron network with three layers.

This network has an input layer (on the left) with three neurons, one hidden layer (in the middle) with three neurons and an output layer (on the right) with three neurons.

There is one neuron in the input layer for each predictor variable. In the case of categorical variables, $N-1$ neurons

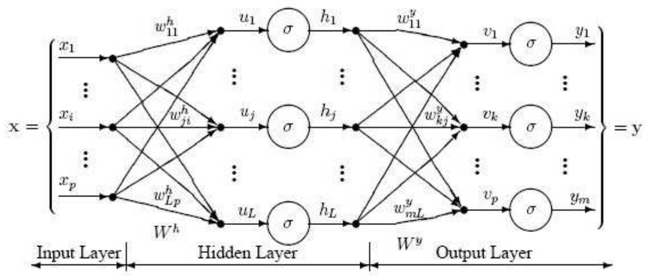


Figure 2: Perceptron network with three layers.

are used to represent the N categories of the variable.

- Input Layer: A vector of predictor variable values ($x_1...x_p$) is presented to the input layer. The input layer (or processing before the input layer) standardizes these values, so that the range of each variable is -1 to 1. The input layer distributes the values to each of the neurons in the hidden layer. In addition to the predictor variables, there is a constant input of 1.0, called the bias that is fed to each of the hidden layers; the bias is multiplied by a weight and added to the sum going into the neuron.
- Hidden Layer: Arriving at a neuron in the hidden layer, the value of each input neuron is multiplied by a weight ($w_j i$), and the resulting weighted values are added, producing a combined value u_j. The weighted sum (u_j) is fed into a transfer function, which generates a value h_j. The outputs from the hidden layer are distributed to the output layer.
- Output Layer: Arriving at a neuron in the output layer, the value from each hidden layer neuron is multiplied by a weight ($w_k j$), and the resulting weighted values are added, producing a combined value v_j. The weighted sum (v_j) is fed into a transfer function, which outputs a value y_k. The y values are the outputs of the network.

If a regression analysis is performed with a continuous target variable, then there is a single neuron in the output layer, and it generates a single y value. For classification problems with categorical target variables, there are N neurons in the output layer producing N values, one for each of the N categories of the target variable.

D. Sequential Minimal Optimization

It is an algorithm described in [22], in which a problem is decomposed successively into two subproblems, decreasing the number of vector operations necessary to resolve the problem. Training a support vector machine requires the solution of a very large quadratic programming (QP) optimization problem. Thus, SMO breaks this large QP problem into a series of smallest possible QP problems. At every step, SMO chooses two Lagrange multipliers to jointly optimize, finds the optimal values for these multipliers, and updates the SVM to reflect the new optimal values.

Two Lagrange multipliers can be done analytically, and this is the main advantage of SMO. Thus, numerical QP optimization is avoided. The inner loop of the algorithm can be expressed using a small code, rather than invoking an entire QP library routine. Also, more optimization sub-problems are solved in the course of the algorithm, each sub-problem is so fast that the overall QP problem is solved quickly. Therefore, SMO requires no extra matrix storage at all. Because no matrix algorithms are used in SMO, the probability to present numerical precision problems is low [22].

SMO is composed by an analytic method for solving for the two Lagrange multipliers, and a heuristic for choosing which multipliers to optimize.

III. Voice Quality Assessment Methodologies

The voice quality assessment is intended to give a score to a particular communication. These results are used to make improvements in transmission networks and, in general, for all the equipment involved in this process.

In this work, Recommendations ITU-T P.862 [3], ITU-T P.563 [4] and ITU-T G.107 [5] will be used in different network scenarios, in which different codecs will be used. The voice quality assessment methodologies and their classification will be shown in the following sub-sections.

A. Methodology Classification

1) Nonintrusive Method: In this type of method, a sample of the original signal communication is not necessary; the evaluation is only determined by the signal at the point the analysis is performed.

The nonintrusive method can be of two types:

- Objective, if one uses a tool (software) for the analysis and calculation of the quality score;
- Subjective, if a listener directly intervenes in the score regarding the quality of the results.

The recommendations below are examples of these methods:

- Objective methods: ITU-T G.107 (E-Model), ITU-T P.563.
- Subjective method: ITU-T P.800 [2].

2) Intrusive Methods: The intrusive method is one that requires a speech sample at the communication origin point, in order to compare with the destination point sample. As result of this comparison a voice quality index is given. Examples of such method are:

- Methods Objectives: Rec. ITU-T P.861 [23] and Rec. ITU-T P.862.

The Figure 3 is adapted from [2] and presents the difference between these two methods.

The following items are described in the ITU-T recommendations P.800, P.862, P.563 and G.107, and these recommendations are used in the experimental tests.

B. ITU-T Recommendation P.800

This recommendation provides a guide to conduct subjective tests of transmission quality in laboratories, and aims to indicate the methods considered appropriate for determining the

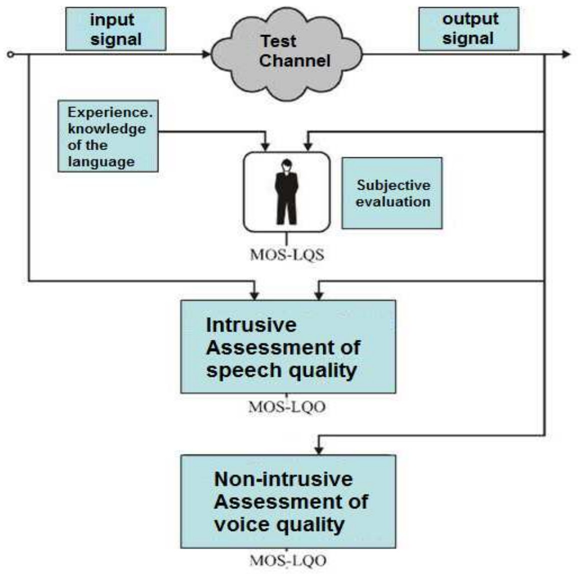


Figure 3: Difference between intrusive and non-intrusive methods [2].

degree of satisfaction by users using a voice communication service.

The recommended methods are:

a) Tests of opinion regarding a conversation.
The conversation tests aim to reproduce in a laboratory the actual conditions of a telephone service. For this, it is necessary to properly select the network conditions and participants, as well as conducting the tests appropriately.
Annex A of recommendation P.800 details the considerations to perform the tests in order to obtain reliable results.
These tests should be performed in a place of dimensions not less than 20 m^3 with a reverberation time smaller than 500 ms, with noise levels that approximate the characteristics of a library or hospital.
In the selection of test participants, the following conditions are required:

- Participants should not have participated directly in the studies of quality evaluation of voice communication services or similar tasks, such as the encoding voice signals.
- The participants should not have participated in any subjective test in the last six months and no conversation tests in the last twelve months.

The ITU-T recommends the following methodologies to be used in the tests:

- Opinion range of the conversation: 5 different scores can be used to assess the various quality categories. The opinion score presented in Table I is the most widely used. In these tests, the listener gives the adjective of the left column and the test lead makes the individual equivalence with the number in the right column. This procedure is valid for other types of tests that are presented in Tables I, II and III.
The arithmetic mean of any set of these scores is called mean opinion of score, which is represented by MOSc (Mean Opinion Score of conversation).
- Difficulty scale: This is a binary response obtained from each participant at the end of each conversation. The question to be performed is: Does the interlocutor or do you experience any difficulty in speaking or listening through the connection? Answer: Yes, No.
Assigning the following values: Yes = 1 and No = 0. The amount assessed (percentage of yes answers) is called the difficulty percentage and is expressed by the symbol: %D.

b) Listening tests - Index determination by Absolute Category Rating (ACR).
As this test does not achieve the same degree of realism as conversation tests, the considerations are less stringent. However, this requires a strict control of certain parameters such as: recording the source voice signal, an adequate calibration of the emitter system, the listening room should have the same conditions as those in a recording room. Regarding the source signals, it is essential to use more than one male voice and one female voice to reduce the risk, since the results depend on the peculiarities of the voices chosen.
The test participants (listeners) are chosen in a population that typically uses telephone services, with the following conditions: not having directly participated in jobs related to the assessment of transmission quality of telephone services, or similar tasks, not having participated in subjective tests for at least six months and listening tests in a year, and having no previous contact with the test sentences.
The opinion scales recommended by the ITU-T for this type of evaluation are:

- Listening-quality scale: the average magnitude of these scores is represented by the symbol MOS. The scores for each category in this scale are presented in Table I.
- Listening effort scale: the average magnitude of these scores is represented by the symbol MOSle. The scores for each category in this scale are presented in Table II.
- Audibility preference scale: the average magnitude of these scores is represented by the symbol MOSlp. The scores for each category in this scale are presented in Table III.
In the ACR method, Annex D from Recommendation ITU-T P.800 defines the DCR

method, which is basically a comparative evaluation between the high-quality original voice and the sample to be evaluated. This method is used when the degradation is small. The scale goes from inaudible to very poor quality. Therefore, it is useful to optimize the system, after ACR method determines that the degradation condition is within the range of acceptable quality.

TABLE I: Score for Conversational MOS and hearing tests

Signal quality	Score
Excelent	5
Good	4
Regular	3
Bad	2
Very bad	1

TABLE II: MOS score for listening effort test

Effort required to understand the meaning of the sentences	Score
Perfect listening, no effort	5
Some attention is necessary, no appreciable effort	4
Moderate effort	3
Considerable effort	2
Meaning incomprehensible, with greater effort	1

TABLE III: MOS score for listening preference test

Audibility Preference	Score
Much greater than preferred	5
Greater than preferred	4
Preferred	3
Smaller than preferred	2
Much smaller than preferred	1

C. ITU-T Recommendation G.107

This recommendation, better known as E-Model [24], is a mathematical and computational model that measures the effects of the network parameters over the voice quality transmitted. The E-Model is defined by the following equation:

$$R = R_o - I_s - I_d - I_e + A \tag{1}$$

In which:

- R: determination factor of the transmission rate that has a correspondence with the MOS score ITU-T P.800.
- R_o: signal to noise ratio.
- I_s: simultaneous degradation factor, which represents all the degradations that occurs simultaneously with the speech signal.
- I_d: quality degradation due to the delay.
- I_e: quality degradation deriving from the device (codec).
- A: improvement factor.

The default value of R_o is 93.2, which is obtained by setting all model variables with default values, for example, the parameter of quality degradation due to delay (I_d), and the parameter corresponding to the equipment degradation (I_e) do not consider the packet loss rate for this calculation. As a result, R_o reaches a high value closer to 100.

The parameter I_s is not considered in this work, since it describes conditions that are related to the signal, not depending on the transport network. The factor A has the value 0 [5] for cable networks, which matches with the emulation scenario used in this work.

The delay factor I_d is defined by (2):

$$I_d = I_{dle} + I_{dte} + I_{dd} \tag{2}$$

Parameters I_{dte} and I_{dle} correspond to delays due to echo, for the sender and receiver, respectively. These factors are not considered for the test scenario assumption of perfect echo suppression. The I_{dd} represents the delay (T_a) produced in both, the codec and the network, respectively. The network delay is set in the network emulator according to the type of test to be performed.

The I_{dd} is defined as:
For $T_a \leq 100ms : I_{dd} = I_d = 0$
For $T_a > 100ms$:

$$I_{dd} = I_d = 0.024d + 0.11(d - 177,3)P(d - 177,3) \tag{3}$$

With: $P(k) = 0$, if $k < 0$, $P(k) = 1$, if $k \geq 0$

With these considerations, the parameter R can be calculated as a function of the parameters that correspond to the value of R_o, the delay (I_d) and the factor corresponding to the codec (I_e).

$$R = R_o - I_d - I_e \tag{4}$$

The R factor is related to the MOS index, according to the following equation:

$$\begin{aligned} R = 3,026 \times M^3 - 25,314 \times M^2 + \\ 87,06 \times M - 57,336 \end{aligned} \tag{5}$$

The relation between R parameter and MOS index is presented in Figure 4 [5].

D. ITU-T Recommendation P.862

The ITU-T Recommendation P.862 [3], known as PESQ is an objective evaluation method that compares an original with a degraded voice signal, resulting from the passage of the voice signal through a communication system. The output is a PESQ quality index, which predicts the perception of quality that would be perceived by an assessor during a subjective listening test. The subjective quality perception is related to a score, called MOS (Mean Opinion Score) index; whereas, the PESQ algorithm estimates the index MOS-LQO and the range used by PESQ goes from 1 (poor) to 4.5 (excellent).

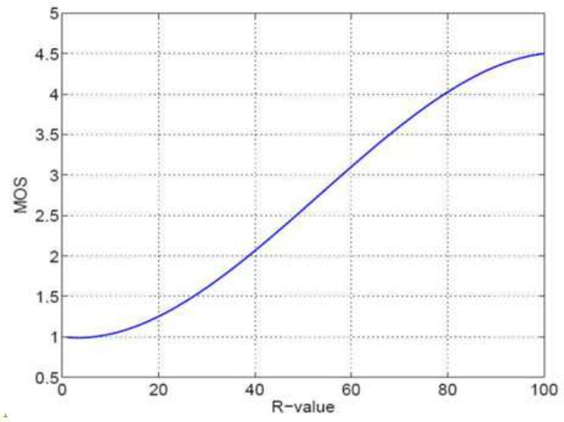


Figure 4: Relation between R parameter and MOS index.

PESQ only measures the effects of one-way speech distortion and noise on speech quality. The effects of delay, echo, and other impairments related to two-way interaction are not reflected in the PESQ scores. According to this recommendation, PESQ demonstrated acceptable accuracy in the following scenarios:

- Speech input levels to a codec.
- Transmission channel errors.
- Packet loss and packet loss concealment with CELP codecs.
- Transcodings.
- Effect of varying delay in listening only tests.
- Short-term and long-term time warping of audio signal.
- Coding Technologies: Waveform codecs, e.g., G.711; G.726; G.727; CELP and hybrid codecs .4 kbit/s, e.g., G.728, G.729, G.723.1; Other codecs: GSM-FR, GSMHR, GSM-EFR, GSM-AMR, CDMA-EVRC, TDMAACELP and TDMA-VSELP.

The G.711 and G.729 codecs were used in the test scenarios, and the network suffers degradation due to packet loss and delay. The audio input was isolated to avoid voice impairments. Only delay is not included in the scenarios of recommendation P.862 tested. To make possible to include this factor in the test scenarios where the delay is present, the MOS index was converted to R index value using equation (5) to obtain the quality index in this type of scenarios.

a) Description of the algorithm used by the PESQ
The PESQ compares an original voice signal with a degraded voice signal, resulting from several types of degradation. The PESQ output is named as MOS-LQO.
In a first step, a series of delays between the original and degraded voice signals are computed, one for each time interval in which the delays are significantly different between the two signals. For each interval, the start and end points are calculated. The alignment algorithm is able to handle delay changes in both periods of silence and in speech.
Figure 5 shows the basic idea used in PESQ. The computational model compares the input and output of the device being tested, comparing the original and degraded output, this degradation is caused by delay or packet loss of the network.
Based on the set of delays encountered, the original and degraded signal already aligned are compared using a perceptual model, as illustrated in Figure 5. The idea of this process is to make both signals into a form of internal representation that is analogous to the psychophysical representation of the signal in the human auditory system, taking into account perceptual frequency and intensity. This is achieved at several stages: time alignment, intensity level alignment, time-frequency mapping, frequency scale transforming, and intensity range compression.

b) Variables that influence the PESQ performance
Tables IV, V and VI show a summary of the factors that must be considered to obtain valid results from the tests.
Table IV shows the test factors, coding technologies and applications in which PESQ has an acceptable performance.
Table V presents some variables in which the PESQ method has unreliable predictions.
Table VI shows factors, technologies and applications for which the PESQ was not evaluated.

TABLE IV: Scenarios in which PESQ algorithm has an acceptable performance

Tests Factors
Input levels of speech signal in a codec.
Errors in the transmission channel.
Packet loss and packet loss concealment with CELP codecs.
Multirate codecs.
Ambient Noise on the sending side.
Coding technology
Waveform codecs, for example: G.711; G.726; G.727.
CELP and hybrid codecs with rate greater than 4 kbit/s,
for example: G.728, G.729, G.723.1.
Other codecs applications: GSM-FR, GSM-HR, GSM-EFR, GSM-AMR,
CDMA-EVRC, TDMA ACELP, TDMA-VSELP.
Aplications
Evaluation of codecs.
Selection of codecs.
Evidence in emulated networks and prototype networks.

TABLE V: Test scenarios unsuitable for PESQ algorithm

Predictions that have no significance
Loss of loudness.
Delay, delay effect in conversational tests.
Eco perceived by the speaker.
In the process of coding with cut voice continuous sections where this period extracted represents more than 25% of the total period of voice activity.
Behavior quality of two-way communications.

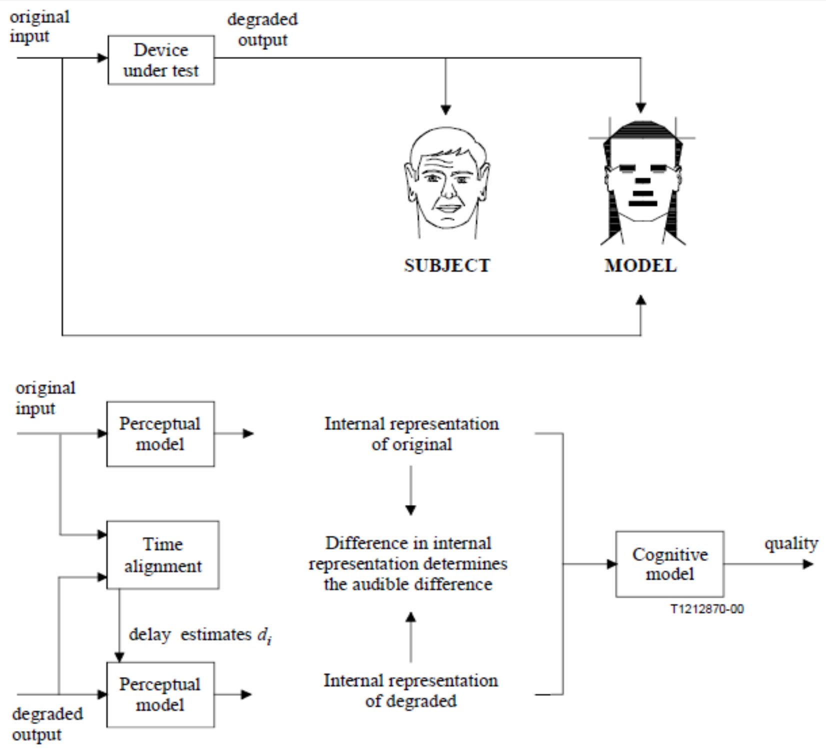


Figure 5: Algorithm used by PESQ [2].

TABLE VI: Test scenarios for which the PESQ has not been evaluated

Testing Factors
Replacement of speaking sessions for silence.
Amplitude saturation of the voice signal.
Effects of dependence with respect to the speaker.
Simultaneous speakers.
Music as codec input.
Eco to the listener and the effects of echo and noise compensators.
Coding technology
CELP and hybrids codecs with rates lower than 4 kbit/s.
MPEG4.
Aplications
Proofs of acoustic terminals through the head and back simulator.

E. ITU-T Recommendation P.563

Algorithm P.563 is applicable to speech quality evaluation without requiring a reference signal. For this reason, it is recommended to a non-intrusive assessment of voice quality and for the supervision of a real network based on only one extreme. This recommendation is not limited to end-to-end measurements, and can also be used at any point in the transmission chain. The score, thus calculated, is comparable to the quality perceived by a human listening to the signal at the test point. This MOS index evaluation is termed MOS-LQO.

This recommendation should be used to assess the speech quality in telephony applications of 3.1 kHz bandwidth. It should be emphasized that algorithm P.563 does not provide a complete assessment of the transmission quality. Thus, it only measures the effects of unidirectional voice distortion and noise in the voice quality, in the same way as a listening test, which assesses the quality in ACR scale. This means that the effects of loss of sound, delay, echo of the speaker, and others that affect the two-way interactions did not influence the P.563 scores. It is worth nothing that this algorithm is designed exclusively for the evaluation of human voice, and can not be applied for music or, generally, other non-vocal audio signals.

The digitized voice signal must meet the following requirements:

- Sampling frequency: 8000 Hz;
- Linear PCM amplitude resolution of 16 bits;
- Minimum duration of 3.0 s and a maximum of 20 s;
- Minimum ratio of vocal activity of 25 % and a maximum of 75 %.

IV. Scenario and Test Methodology

The tests were conducted in the emulation of IP network shown in Figure 6, which consists of three computers; two of them (PC-A and PC-C) establish point to point VoIP communication, and the third (PC-B) emulates the transmission channel, which programmed different degradations of network, such as packet loss and delay. The audio inputs used belong to the set of validation tests that are included in ITU-T P.862, the algorithm of the same recommendation (PESQ) is used to assess the level of quality of voice signal in reception.

The second quality index is determined by machine learning algorithm; these algorithms are decision trees, neural networks, SMO and Bayesian networks. The first step is to build the algorithm used in machine learning or training file, which was built considering 650 scenarios, where QoS was obtained from the E-model algorithm. Each test scenario is represented in the training file as a line, which will be presented later in more detail. In the emulator network, different scenarios are programmed, with different parameter values of probability of packet loss and delay inserted in the training file in order to better validate the reliability of results. The sample period for assessing the quality of VoIP communications is 8 seconds. This time was chosen, in addition to the original size of the audio file, due to the number of packets sent by the transmitter (PC-A) during this time period. Thus, considering an encoding rate of 64 kbps, 400 packets of 160 bytes are sent; for a shorter time, the number of packets sent is smaller and, therefore, the percentage of packet loss has a lower resolution.

The software and tools, in each PC used to build the test scenario are the following:

- PC-A, PC-C: clients using softphone MyPhone 0.2b10 [25], packet analyser Wireshark [26] and software to record audio, a .wav file,VRS Recording System [25].
- PC-B: router using the network emulator NETEM [28], to simulate packet loss, delay, jitter and bandwidth; software ITU-T P.862 was used to find the MOS index for voice quality.

The sound transmitted was generated by a player of a .wav file that is connected to the microphone input of PC1 by an audio cable. As mentioned previously, this file has a duration of 8 seconds and was sampled at 8 kHz and 16 bits.

The methodology followed in the tests to get the value using the tool PESQ MOS is as follows:

- Initially, it starts a communication between the PC-A and PC-C by softphones installed in the PCs, and lets you choose the voice codec used to each VoIP call, made from computer to computer.
- For each scenario, the network emulator is configured with parameters required to perform the test.
- The player transmits the audio (arq-orig.wav) to PC-A where this sound is recorded (arq-orig2.wav), as the call is active, and the voice is transmitted to the PC-C through the PC-B.
- While data are transmitted, the program Wireshark running in PC-A and PC-C saves network information, such as the signaling messages for establishing, maintaining and finalizing calls, messages from the RTP, the average size of the package (bytes), average number of packets transferred per second and average bandwidth.
- In the PC-C, the audio received is recorded in a file (arq-deg.wav). This file and the original file are compared by software PESQ, which runs on PC-B, resulting in a MOS-LQO score.

A. Implementation of the input file for training algorithms

As aforementioned, the preparation of an initial database of network parameters and the quality score for each scenario was performed considering the ITU-T Recommendation G.107. This mathematical model provides an R value of quality ranging from 0 to 93.2, in which the highest value corresponds to a higher quality. The different scenarios were done leaving the default parameters fixed and varying the following parameters:

- The Mean One-Way Delay.
- The Packet-loss Probability.
- The type of codec is related to the parameters: encoding rate, I_e (Equipment Impairment Factor) and Bpl (Packet-loss Robustness Factor).

The values of I_e and Bpl are dependent on the vocoder used. Table VII presents the values of these codec parameteres that have been tested by the ITU-T recommendation G.107. In our test scenarios, codecs G.711 and G.729 were employed.

TABLE VII: Values of I_e and Bpl for voice codecs

Codec	I_e	Bpl
G.723.1+VAD	15	16.12
G.729A+VAD	11	19
GSM-EFR	5	10.03
G.711	0	4.3
G.711+PLC	0	25.14

The parameters described in Table VIII were used to obtain the value of quality index R.

TABLE VIII: Parameters of the E-Model Algorithm

Parameter	Default Value	Units
Noise Referred to at 0 dBr point - Nc	-70	dBm
Noise Floor - Nfor	-64	dBm
Room Noise (Receive) - Pr	35	dB
Send Loudness Rating - SLR	8	dB
D-factor (Receive) - Dr	3	-
Listener's Sidetone Rating - LSTR	21	dB
D-factor (Send)	3	-
Mean One-Way Delay - T	100	ms
Absolute Delay from (S) to (R) - Ta	100	ms
Round-Trip Delay - Tr	200	ms
Weighted Echo Path Loss - WEPL	110	dB
Quantizing Distortion Units - QDU	1	-
Equipment Impairment Factor - Ie	0	-
Packet-loss Robustness Factor - Bpl	1	-
Packet-loss Probability - Ppl	1	%
Expectation Factor - A	0	-

As a result, a file with 650 lines was obtained; for better understanding, Table IX presents the 10 first cases or network scenarios.

TABLE IX: Values of I_e and Bpl for voice codecs

Rate (kbps)	Delay (ms)	Bpl (%)	Ie	R (Value)
64	0	4.3	0	93.2
64	50	4.3	0	91.8
64	100	4.3	0	90.7
64	150	4.3	0	89.5
64	200	4.3	0	85.8
64	250	4.3	0	79.2
64	300	4.3	0	72.5
64	350	4.3	0	67
64	400	4.3	0	62.2
64	450	4.3	0	58.2

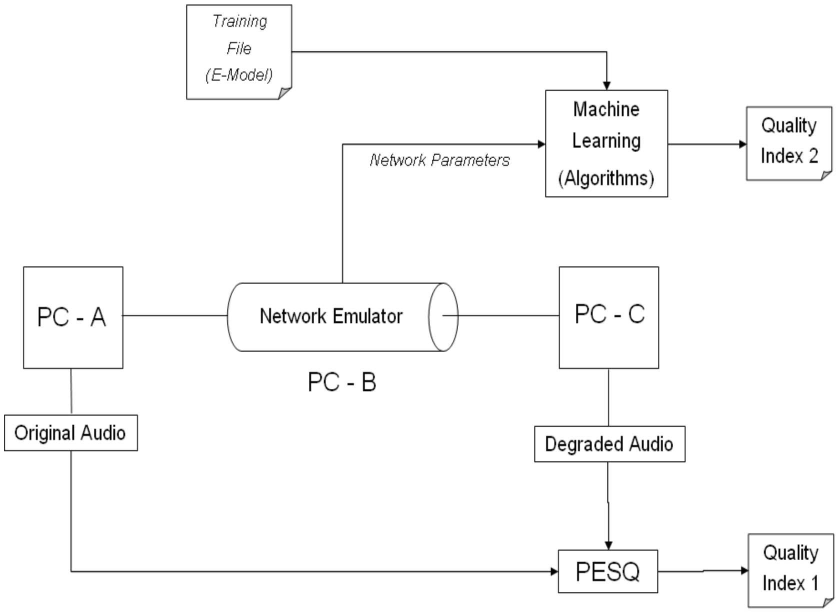


Figure 6: Test scenario.

In order to group ranges of R values and define QoS categories, we took the as reference model the classification presented by ITU-T Recommendation G.107. This classification model is presented in Figure 7, with the limit values for both R and MOS index.

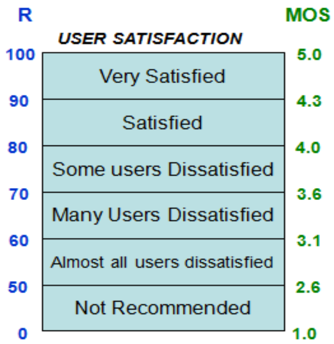


Figure 7: Levels of User Satisfaction of a VoIP communication.

The classification quality levels used herein only consider five categories; the last two categories, *Nearly All Users Dissatisfied* and *Not Recommended* in Figure 6, were grouped into one. These five categories are presented in Table X.

This categorization allows obtaining the file that will work as a training file to the algorithms that determines the quality of the communication. Table XI presents ten sample lines from the training file.

TABLE X: QoS levels used in the test scenario

Categories	R (Min. value)	R (Max. Value)
Very Satisfied (QoS_1)	90	94
Satisfied (QoS_2)	80	90
Some Users dissatisfied (QoS_3)	70	80
Many Users dissatisfied (QoS_4)	60	70
Nearly all Users Dissatisfied and Not Recommended (QoS_5)	0	60

TABLE XI: Codec and network parameters and the resulting QoS

Rate (kbps)	Delay (ms)	Bpl (%)	QoS
64	0	0	QoS_1
64	50	0	QoS_1
64	100	0	QoS_1
64	150	0	QoS_1
64	200	0	QoS_2
64	250	0	QoS_2
64	300	0	QoS_3
64	350	0	QoS_4
64	400	0	QoS_4
64	450	0	QoS_5

B. Determination of QoS using the Weka tool

In order to determine the QoS using the learning algorithms, Software Weka-version 3.7.4 [29] was used as a tool for data analysis method. This tool supports several algorithms, based on related works. The following four algorithms were used in the tests:

- Desicion Tree J48 - algorithm C4.5;
- Bayesian networks - Naive Bayes;
- Neural Networks - Multilayer Perceptron.
- Sequential Minimal Optimization - SMO.

With the training file, the cross-validation was analyzed and the values of the factor F (F-measure) for each algorithm tested are shown in Table XII.

TABLE XII: Values of F-measure for each algorithm and QoS

Algorithm	QoS-1	QoS-2	QoS-3	QoS-4	QoS-5
Trees J.48	0.99	0.96	0.96	0.98	0.97
Multilayer Perceptron	0.94	0.95	0.97	0.94	0.96
SMO	0.87	0.90	0.89	0.84	0.94
Bayes (Naives)	0.90	0.88	0.89	0.81	0.86

The values reached for the F factor (F-measure) were very high, whereas values greater than 0.7 are enough for network training. The decision tree algorithm obtained the best results, with higher F-measure value.

V. Experimental Results

In this work, 600 tests were conducted following the methodology explained. Each test considered a scenario configured with different parameters of network emulator. The larger the number of tests in learning techniques, the greater the validity of the results.

The results are presented in Figure 8, which depicts the highest value of success in determining the quality of service of VoIP communication for each learning algorithm used; the Decision Trees algorithm reaches 589 valid test results, that means, 589 results are concordant with the results obtained from PESQ. Also, the Multilayer Perceptron reached a high accuracy value with 570 satisfactory test results. SMO and Bayes algorithms reached 519 and 504 valid results, respectively. It is important to note that the performances of these four algorithms are concordant with the F-measure values presented in Table XII.

It worth noting that these results were obtained considering the range of MOS values of each QoS category and not a single value of index MOS.

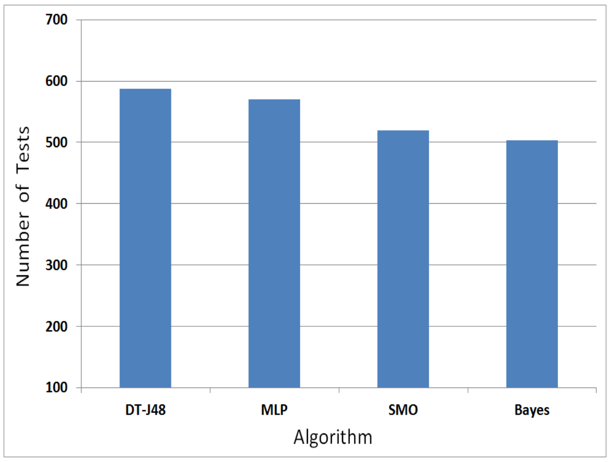


Figure 8: Algorithm results - Number of satisfactory test results according to PESQ algorithm.

VI. Conclusions and Future Work

The test results show that machine learning is a valid method to determine the quality of a VoIP communication in several network conditions and using different voice codecs, for this work, codecs G.711 and G.729. The good performance of the learning algorithms definitely depends on the initial file used to train these algorithms.

In this work, parameter values from a real IP network were used, and the test was performed in network with realistic parameters. Also, the confidence of the E-Model algorithm was tested to determine the values of the quality index for each scenario included in the training file. The best result was obtained by the Decision Tree algorithm that reached 98% of accuracy, which is a very high degree of reliability to determine the quality category of VoIP communication in relation to other works. The Multilayer Perceptron algorithm had a high level of success, too, reaching 95% accuracy. The Naive Bayes algorithm and SMO reached 86.5% and 84% of accuracy, respectively. The tests were performed in an emulated network with free softwares. For this reason, the implementation of the same scenario in other works is possible.

As future work, we intend to evaluate the quality of video services, for example, streaming video, creating a training file based on ITU-T Recommendation P.930 [30]. Also, different techniques of network resource allocation will be studied based on predictions of quality services of a future period of time, the goal of this idea being to improve the quality of a specific service.

Acknowledgments

The authors thanks University of São Paulo for the motivation to researches in the area of Computer and Telecommunication Systems. This work was supported by FAPESP (The State of São Paulo Research Foundation - Brazil). FAPESP project number: 2011/12724-8.

References

[1] D. Z. Rodríguez, L. R. Rosa, and G. Bressan, *Predicting the Quality Level of a VoIP Communication through Intelligent Learning Techniques*. The Seventh International Conference on Digital Society (ICDS), pp. 42-47, Nice, France, 2013.

[2] ITU-T Rec. P.800, *Methods for subjective determination of transmission quality*. Aug. 1996. Disponível em: www.itu.int/rec/T-REC-P.800/en, 08.12.2013.

[3] ITU-T Rec. P.862, *Perceptual evaluation of speech quality (PESQ): An objective method for end-to-end speech quality assessment of narrowband telephone networks and speech codecs*. http://www.itu.int/rec/TREC-P.862/en, 08.12.2013.

[4] ITU-T Rec. P.563, *Single-ended method for objective speech quality assessment in narrow-band telephony applications*, May. 2004. Disponível em: www.itu.int/rec/T-REC-P.563/en, 08.12.2013.

[5] ITU-T Rec. G.I07, *The E-model, a computational model for use in transmission planning*, International Telecommunications Union, Mar. 2005.

[6] L. A. R. Yamamoto and J. G. Beerends, *The impact of network performance parameters on the end-to-end perceived speech quality*, Expert ATM Traffic Symposium, Sep. 1997.

[7] S. R. Broom, *VoIP quality assessment: tracking account of the edge device*, IEEE Trans. on Audio, Speech and Language Processing, vol. 14, pp. 1977-1983, Nov. 2006.

[8] R. Eisinger. *Adaptação dinâmica de vídeo*. Master's thesis, Apr. 2007.

[9] A. Doulamis and G. Tziritas. *Content-based video adaptation in Low/Variable bandwidth communication networks using adaptable neural network structures*. In The 2006 IEEE International Joint Conference on Neural Network Proceedings, pages 4037-4044, Vancouver, BC, Canada, 2006.

[10] L. Sun and E. C. Ifeachor, *Perceived speech quality prediction for voice over IP based networks*, IEEE International Conference on Communications, vol. 4, pp. 2573-2577, Apr. 2002.

[11] S. Mohamed, F. Cervantes-Perez, and H. Afifi, *Integrating networks measurements and speech quality subjective scores for control purposes*, INFOCOM 2001 Proceedings, vol. 2, pp. 641-649, Apr. 2001.

[12] G. Corrigan, N. Massey, and O. Schnurr, *Transition-based speech synthesis using neural networks*, ICASSP 2000 Proceedings, vol. 2, pp. 945-948, Jun. 2000.

[13] M. M. Meky and T. N. Saadawi, *Prediction of speech quality using radial basis functions neural networks*, IEEE Proceeding on Computers and Communications, pp. 174-178, Jul. 1997.

[14] ITU-T Rec. G.711, *General Aspects of Digital Transmission Systems Terminal Equipments - Pulse Code Modulation (PCM) of Voice Frequencies*. 1972. http://www.itu.int/rec/T-REC-G.711/en, 08.12.2013.

[15] ITU-T Rec. G.729, *Coding of speech at 8 kbit/s using conjugate-structure algebraic-code-excited linear prediction (CS-ACELP)*, Janeiro 2007. http://www.itu.int/rec/T-REC-G.729/en, 08.12.2013.

[16] S. Lingfen, *Perceived speech quality prediction for voice over IP-based networks*, IEEE International Conference on Communications, vol. 4, pp. 2573-2577, 2002.

[17] R. Jiuchun, D. Mao, and Z. Wang, *A neural network based model for VoIP speech quality prediction*, Proceedings of the 2nd International Conference on Interaction Sciences, pp. 1244-1248, 2009.

[18] J. R. Quinlan, *Induction of decision trees, Machine Learning*, 1986, vol. 1, issue 1, pp. 81 - 106.

[19] J. Stutz, *Bayesian Classification Theory*, Techinical Report, 1991.

[20] F. Rosenblatt, *Perceptron Simulation Experiments*, Proceedings of the IRE, vol. 48, pp. 301 - 309, 1960.

[21] S.-I. Horikawa, *On fuzzy modeling using fuzzy neural networks with the back-propagation algorithm*, IEEE Transactions on Neural Networks, vol. 3, pp. 801 - 806, 1992.

[22] J. C. Platt, *Advances in kernel methods*. Cambridge, MA, USA: MIT Press, 1999.

[23] ITU-T Rec. P.861, *Objective quality measurement of telephone-band*. http://www.itu.int/rec/TREC-P.861/en, 08.12.2013.

[24] L. Ding, *Speech quality prediction in VoIP using the extended E-model*, IEEE Global Telecommunications Conference, pp. 3974 - 3978, vol.7, 2003.

[25] *Softphone Myphone*, http://myphone.sourceforge.net/, 08.12.2013.

[26] *Wireshark*, http://www.wireshark.org/download.html, 08.12.2013.

[27] *VRS - Recording System*, http://www.nch.com.au/vrs/index.html, 08.12.2013.

[28] *NETEM - Network Emulator*, http://www.linuxfoundation.org/collaborate/workgroups/networking /netem, 08.12.2013.

[29] Weka, *Weka 3: Data Mining Software in Java*. http://www.cs.waikato.ac.nz/ml/weka/, 08.12.2013.

[30] ITU-T Recommendation-P.930, *Principles of a reference impairment system for video*, Geneva, Aug. 1996.

Easy Development of Web Applications using WebODRA2 and a Dedicated IDE

Mariusz Trzaska
Chair of Software Engineering
Polish-Japanese Institute of Information Technology
Warsaw, Poland
mtrzaska@pjwstk.edu.pl

Abstract - **The modern Web requires new ways for creating applications. We present our approach combining a web framework with a modern object-oriented database and a dedicated Integrated Development Environment (IDE). It makes it easier to develop web applications by rising the level of abstraction. In contrast to many existing solutions, where the business logic is developed in an object-oriented programming language and data is stored and processed in a relational system, our proposal employs a single programming and query language. Such a solution, together with flexible routing rules, creates a coherent ecosystem and, as an additional benefit, reduces the impedance mismatch. Our research is supported by a working prototype of the IDE and a web framework for our own object-oriented database management system. Furthermore, the created IDE utilizes scaffolding, which can automatically generate web GUIs supporting some useful operations.**

Keywords-Web frameworks; Web tools; Web applications; Object-Oriented Databases; Integration Development Environment; IDE; DSL editors; Scaffolding

I. Introduction

Web frameworks are commonly utilized in software development. Moreover, it seems that for each popular programming language exists at least a few different proposals. The situation is different in case of prototype technologies. In [1], we have presented our proposal of a web framework dedicated to the object-oriented database ODRA (Object Database for Rapid Application development).

It seems that the most successful frameworks (e.g., Rails, ASP.NET MVC) follow the three-tier architecture: a presentation layer, business logic (a middle tier) and a data tier. Each of them can be developed through a different technology and can utilize incompatible data models.

Typically, the middle tier is developed using an object-oriented programming language such as Java, MS C#, Ruby, etc. However, the object-orientedness is a bit blurry concept. There is no single, well-accepted, specific definition or set of properties, which determine features of an object-oriented programming language. Java and C# are pretty close to each other in that area, but for instance Ruby is based on quite different concepts.

Contrary to implementation of the business logic, the data is usually stored using a relational database system. This causes a negative phenomenon known as impedance mismatch. Our framework has been created not only as an aid for making websites but also as an attempt to remove the fault. Of course, during the years, numerous approaches have been proposed to solve or reduce the problem. In particular, following Trzaska [2], the solution could use a single model both for the business logic and data.

Aside of frameworks, one of the most popular software, widely utilized by programmers, is an Integrated Development Environment (IDE). Various IDEs are on the scene for many years. They provide many different services and are invaluable help during software development. At the basic level they just support a programming language. However, their real power can be experienced when they have dedicated functionalities for particular frameworks. Similarly to the situations with the frameworks, prototype solutions are rarely equipped with an IDE.

In this paper, we would like to employ the idea for a tool aiming at creating web applications. We propose a paradigm, which uses the same high level language for working with data and implementing a business logic. In fact, those two utilizations are indistinguishable.

On the software level, our solution consists of two parts:

- An Integrated development Environment (called ODRA eIDE2) optimized for the ODRA DBMS, SBQL (Stack-Based Query Language) language and WebODRA2;
- A new version of the web framework (presented in [1]) called WebODRA2, which integrates two independent components:
    - The object-oriented DBMS ODRA with SBQL, a powerful programming and query language;
    - A web server.

This approach increases significantly the level of abstraction, which reduces the implementation time, decreases the number of errors and of course, completely eliminates the impedance mismatch. The programmers are able to focus on website's creation using a single, coherent technology.

The main contribution of the paper are the following:

- A new coherent paradigm of creating web application using the same high level programming and query language;
- A working prototype implementation of the approach containing a dedicated IDE, object-oriented database, web server and all the necessary components.

The rest of the paper is organized as follows. To fully understand our motivation and approach, some related solutions are presented in Section II. Section III briefly discusses key concepts of the utilized database and programming/query language. Section IV presents the prototype implementation of the proposed IDE and the web framework. Section V is devoted to the scaffolding mechanism. Section VI concludes.

II. Related Solutions

The related solutions appropriate for this paper could be analyzed from two points of view: web frameworks and IDEs. The next two sections contain their discussion.

A. *Existing Web Frameworks*

There are a lot of different web frameworks that use various approaches; just to name the most popular ones (by platform):

- Java: Apache Struts, Java Server Faces, JBoss Seam, Spring, Grails (Groovy), Play (Scala and Java);
- MS C#: ASP.Net, ASP.NET MVC, Kentico;
- PHP: CakePHP, Symfony, Zend;
- Smalltalk: Seaside [3];
- Ruby: Ruby on Rails, Sinatra.

They differ in details but unfortunately share the same problems related to inconsistent data models for programming languages and data. Even when an object-relational mapper is utilized the impedance mismatch problem is decreased, but not removed. For instance, the Ruby's Active Record requires additional information from a programmer to specify some non-mappable objects like arrays [4].

However, it is also possible to find solutions, where a website is developed using a single model. The next paragraphs contain description of such frameworks.

CouchApp [5] is a technology, which allows for creating applications delivered to the browser from CouchDB [6]. Applications are implemented using JavaScript and HTML5. The general idea is quite similar to our approach because CouchDB is a database management system. However, on contrary to our framework, the DBMS follows the NoSQL philosophy and allows one to store documents in the JSON [7] format. There is also no query language similar to SQL or our SBQL (see Section III). All database queries are performed using dedicated API and JavaScript. The result is also returned as a JSON data.

Every web application needs a GUI. In case of CouchApp a GUI is created as a transformation of returned JSON data into some other format. For instance there are functions, which together with dedicated views, are able to convert the data into HTML, XML, CVS, etc.

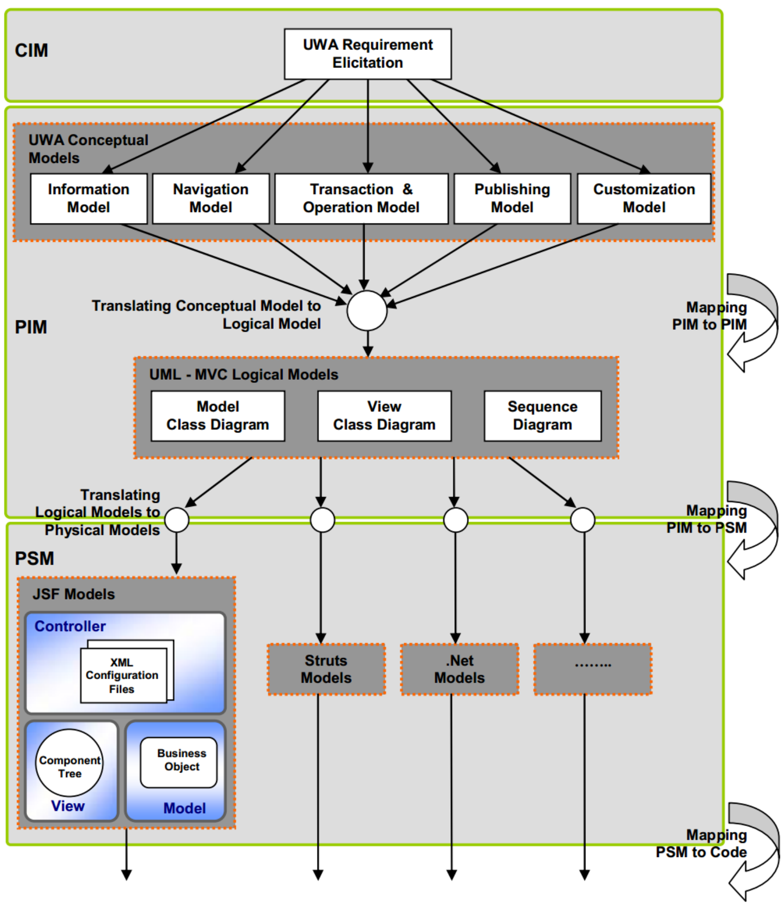


Figure 1. An overview of the UWA-based MDD process. Source: [8].

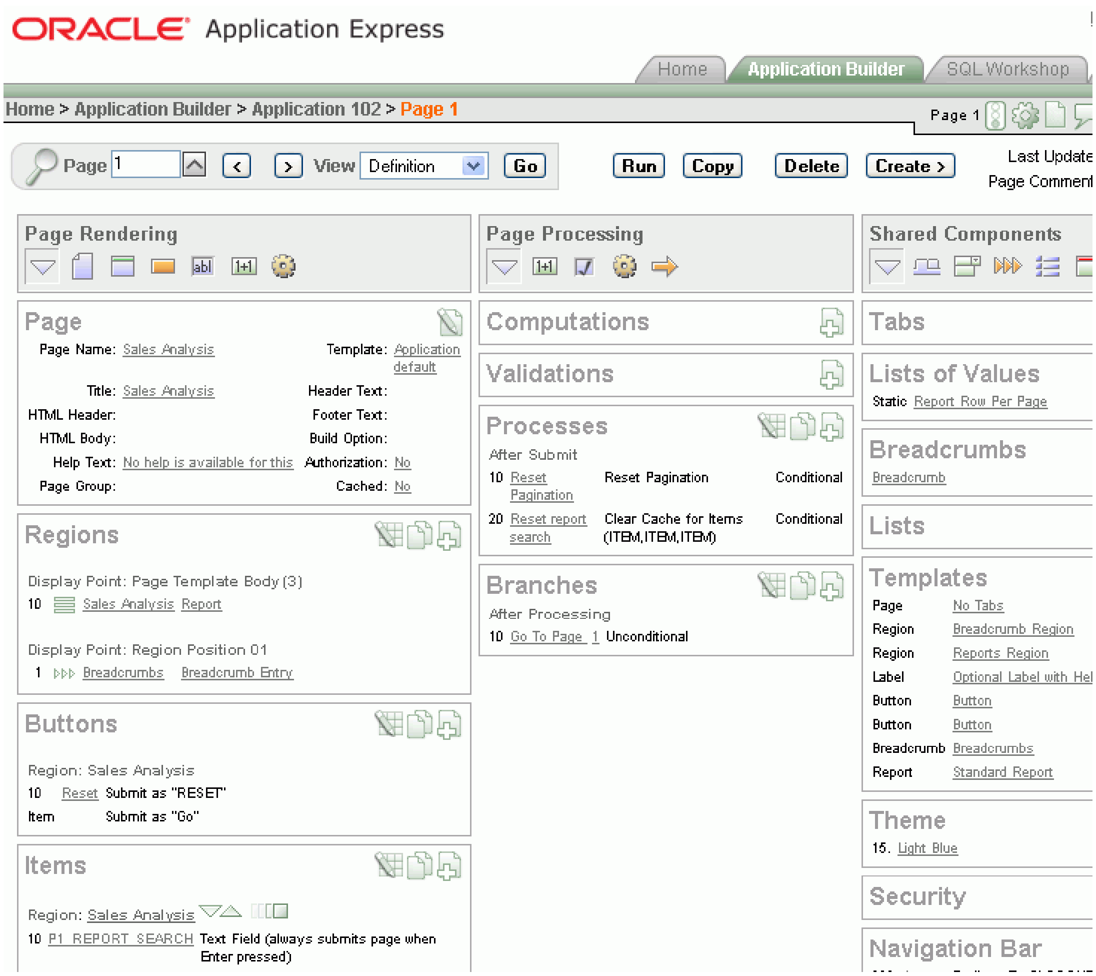


Figure 2. Oracle Application Express. Source: [9]

Another approach to create a web application might employ the Model Driven Architecture (MDA) paradigm. The idea is to define a model (or models) and, through some transformations, receive a working application. There is a lot of such systems, see for instance [10], [11], [8]. However, they are not widely utilized. One of the reason could be the amount and type of work, which has to be done to get a working website. For instance, the proposal presented in [8], which is quite common to all MDA solutions, needs the following models and information to be precisely defined (Figure 1):

- UWA requirements,
- Information model,
- Navigation model,
- Transaction & operation model,
- Publishing model,
- Customization model,
- Logical models (UML diagrams): class, sequence.

Of course, the above information is not only required by MDA tools. Furthermore, they have to be provided by all websites' developers. It seems that the way of defining them makes the difference in popularity.

The last described solution is not exactly a framework for programmers. Oracle Application Express (Figure 2) [12] is more like a tool for a rapid web application development for the Oracle database. It is available, under different names, since 2000. The application requires a dedicated server and provides easy-to-use programming environment accessible via a web browser.

Most of its functionalities are available through dedicated graphical user interfaces, various wizards and helpers. But, still there are possibilities for using a programming language, namely PL/SQL. SQL, despite of thirty-year existence and big popularity, is the subject of heavy criticism. The SQL's flaws like: inconsistencies, incompatibilities between vendors and shortcomings of the relational model, decrease the value of solutions. Furthermore, application generators have some inherent shortcomings, which make their products less flexible (in terms of usability, functionality, GUI) than applications developed by programmers. We believe that using a more powerful programming and query language, together with an object-oriented model, presents a much better approach.

B. The Most Popular IDEs

Below we present the most popular IDEs for particular programming languages/platforms:

- Java: Eclipse [13], NetBeans [14], IntelliJIDEA [15];
- MS C#: MS Visual Studio [16], MonoDevelop [17], SharpDevelop [18];
- PHP: Aptana Studio [19], Eclipse PDT [20], PhpStorm [21], KDevelop [22];
- Smalltalk: Pharo [23], VisualWorks [24];
- Ruby: RubyMine [25], NetBeans [26], Aptana RadRails [27].

The IDEs differ in many ways. They can be distributed completely for free (e.g., Eclipse, NetBeans) or have various (usually limited) editions, e.g., Visual Studio: Professional Edition costs a few hundreds of USD whereas Express Edition is free.

Some of them are strictly dedicated to particular language/technology e.g., RubyMine. Others support various programming languages – sometimes using special plugins (e.g., Eclipse: Java, C++, and PHP).

Concerning functionality, it seems that the basic possibilities are quite similar in all IDEs and include:

- Syntax coloring (see Figure 3). Text of a program (source code) is presented using different colors, accents (e.g., bold) and decorators (e.g., an underline);
- Autocomplete. This is one of the most important features of a decent IDE and shows a list of possibilities for a particular context (e.g., after the dot the programmer sees attributes of a class);
- Error reporting. Errors and warnings are presented in a special window and sometimes inside the editor as well;
- Quick fix. When an error or a warning is shown to a user, he/she can choose one of proposed fixes to the problem (e.g., adding an import statement in case of an unknown class);
- Semantic navigation among artefacts. This functionality allows jumping from an occurrence of an element to its definition, e.g., from an object to its class;
- References. This makes possible finding all utilizations of particular artefacts (e.g., all method calls);
- Project Management. Takes care of all project's files including sources, assets, folders, etc. Usually it is possible to connect it to a versioning management system (e.g., Subversion, Git, etc.);
- Plugins. They provide a way for adding new functionalities (e.g., support for a new framework and/or programming language) to existing core. Sometimes (e.g., Eclipse) an entire IDE is based on plugins.
- Refactoring. This technology is responsible for making changes to a source code without modifying its behavior. There are dozens of different refactoring, e.g., renaming, extracting an interface or a method, adding getters/setters, etc.;

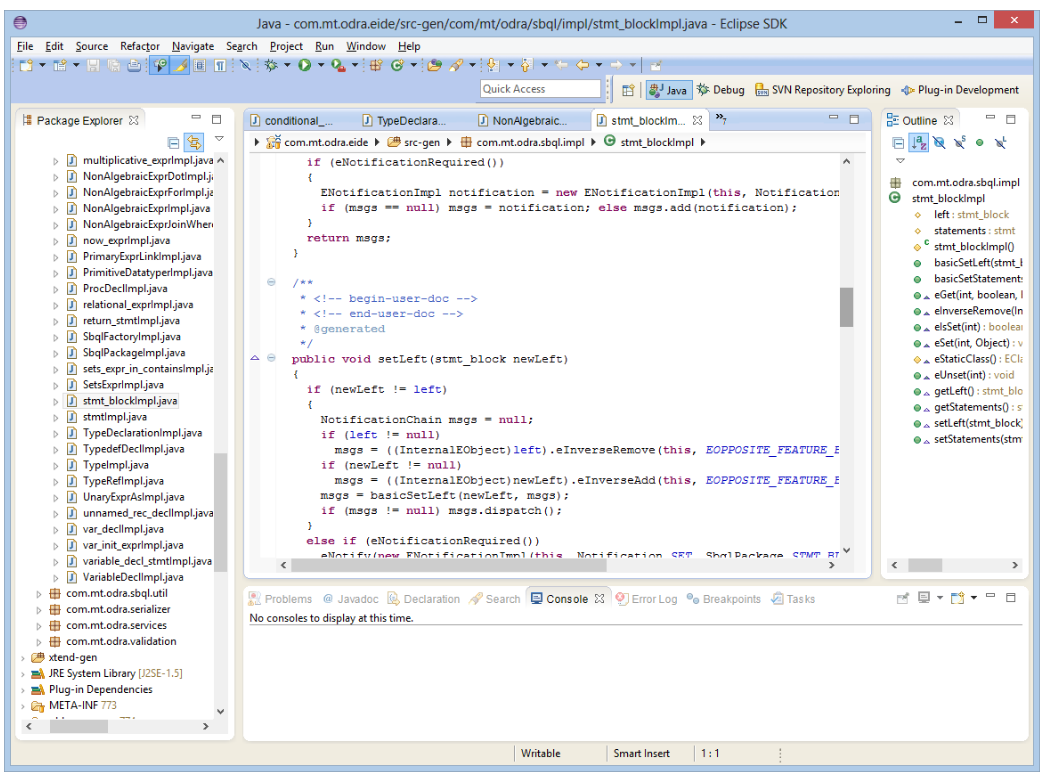


Figure 3. The Eclipse IDE. Source: own elaboration.

- Analyzing the code. In some cases, IDE is able to make suggestions regarding a source code, e.g., renaming an artefact according to a particular convention.
- Scaffolding. This is a mechanism, which helps programmers and/or designers in creating a starting point for further development. It could generate various artefacts, but in most cases is utilized for a GUI creation. It is especially popular in web frameworks where the mechanism is able to generate necessary files (HTML, CSS, etc.) required for CRUD (Create, Retrieve, Update, Delete) operations. Sometimes it is hard to distinguish if this tool is integrated with a framework or it is a separate component embedded in IDE.

In many cases, modern IDEs are equipped with GUI editors (embedded or installed through additional plugins). Their functionalities vary starting from simple generators to sophisticated bi-directional masterpieces.

Modern software projects are usually implemented using a wide range of technologies, platforms and frameworks. Thus support for them is essential. This is an area where commercial products show their strengths (e.g., IntelliJIDEA).

The choice of a particular IDE is not always based solely on its functionalities. In some projects also non-functional requirements could be important, e.g., cross-platform support. This is satisfied by most of the IDEs because many of them is developed using cross-platform technologies like Java (Eclipse, NetBeans, Aptana Studio).

It is also worth noting “smart” editors (Notepad++, Vim, Sublime Text, and TextMate), which could be very close to fully-fledged IDEs, especially after a correct configuration. They usually offer less sophisticated functionalities like syntax coloring or project’s files management.

III. THE ODRA DATABASE

As previously mentioned, our proposal for creating websites is based on utilization of an object-oriented database together with a powerful query and programming language. DBMS could be used as a data storage and could be utilized to implement business logic. For the purpose of the first requirement we need a database query language. However, because of the second necessity, we might need something more flexible and powerful: a fully-fledged programming language with imperative constructs. Both criteria are met by our prototype DBMS called ODRA.

ODRA is a prototype object-oriented database management system [28], [29], [30], [31] based on SBA (Stack-Based Architecture) [16]. The ODRA project started to develop new paradigms of database application development. This goal is going to be reached mainly by increasing the level of abstraction, at which the programmer works. ODRA introduces a new universal declarative query and programming language SBQL [28], together with distributed, database-oriented and object-oriented execution environment. Such an approach provides functionality common to the variety of popular technologies (such as relational/object databases, several types of middleware, general purpose programming languages and their execution environments) in a single universal, easy to learn, interoperable and effective to use application programming environment.

ODRA consists of three closely integrated components:

- Object Database Management System (ODMS),
- Compiler and interpreter for object-oriented query programming language SBQL,
- Middleware with distributed communication facilities based on the distributed databases technologies.

The system is additionally equipped with a set of tools for integrating heterogeneous legacy data sources. The continuously extended toolset includes importers (filters) and/or wrappers to XML, RDF, relational data, web services, etc.

ODRA has all chances to achieve high availability and high scalability because it is a main memory database system with memory mapping files and makes no limitations concerning the number of servers working in parallel. In ODRA many advanced optimization methods that improve the overall performance without compromising universality and genericity of programming interfaces have been implemented.

The next subsections contain a short discussion of the ODRA main features including its query and programming language SBQL.

A. ODRA Object-Oriented Data Model

The ODRA data model is similar to the UML object model. Because in general UML is designed for modeling rather than for programming several changes have been made to the UML object model that do not undermine seamless transition from a UML class diagram to an ODRA database schema. The ODRA object model covers also the relational model as a particular case; this feature is essential for making wrappers to external sources stored in relational databases. Below, we present a short description of the main data model elements:

- Objects. The basic concept of the ODRA database model is object. It is an encapsulated data structure storing some consistent bulk of information that can be manipulated as a whole. A database designer and programmers can create database and programming objects according to their own needs and concepts. Objects can be organized as hierarchical data structures, with attributes, sub-attributes, etc.; the number of object hierarchy levels is unlimited. Any component of an object is considered an object too.
- Collections. Objects within a collection have the same name; the name is the only indicator that they belong to the same collection. Usually, objects from a collection have the same type, but this requirement is relaxed for some kinds of heterogeneous collections. Collections can be nested within objects with no limits (e.g., in this way it is possible to represent repeating attributes).
- Links. Objects can be connected by pointer links. Pointer links represent the notion that is known from UML as association. Pointer links support only binary associations; associations with higher arity and/or

with association classes are to be represented as objects and some set of binary associations. This is a minor limitation in comparison to UML class diagrams, introduced to simplify the programming interface. Pointer links can be organized into bidirectional pointers enabling navigation in both directions.

- Modules. In ODRA, the basic unit of database organization is a module. As in popular object-oriented languages, a module is a separate system component. An ODRA module groups a set of database objects and compiled programs and can be a base for reuse and separation of programmers' workspaces. From the technical point of view and of the assumed object relativism principle, modules can be perceived as special purpose complex objects that store data and metadata.
- Types, classes and schemata. A class is a programming abstraction that stores invariant properties of objects, in particular, its type, some behavior (methods, operations) and (optionally) an object name. A class has some number of member objects. During processing of a member object the programmer can use all properties stored within its class. The model introduces atomic types (integer, real, string, date, boolean) that are known from other programming languages. Further atomic types are considered. The programmer can also define his/her own complex types. Collection types are specified by cardinality numbers, for instance, [0..*], [1..*], [0..1], etc.
- Inheritance and polymorphism. As in the UML object model, classes inherit properties of their superclasses. Multiple inheritance is allowed, but name conflicts are not automatically resolved. The methods from a class hierarchy can be overridden. An abstract method can be instantiated differently in different specialized classes (due to late binding); this feature is known as polymorphism.
- Persistence and object-oriented principles. The model follows the orthogonal persistence principle, i.e., a member of any class can be persistent or volatile. Shared server objects are considered persistent, however, non-shared objects of a particular applications can be persistent too. The model follows the classical compositionality, substitutability and open-close principles assumed by majority of object-oriented programming languages.

Distinction between proper data and metadata (ontology) is not the property of the ODRA database model. The distinction can be important on the business model level, but from the point of view of ODRA both kinds of resources are treated uniformly.

B. *Query and Programming Language SBQL*

SBQL (Stack-Based Query Language) is a powerful query and programming language addressing the object model described above. SBQL is precise with respect to the specification of semantics. SBQL has also been carefully designed from the pragmatic (practical) point of view. The pragmatic quality of SBQL is achieved by orthogonality of introduced data/object constructors, orthogonality of all the language constructs, object relativism, orthogonal persistence, typing safety, introducing all the classical and some new programming abstractions (procedures, functions, modules, types, classes, methods, views, etc.) and following commonly accepted programming languages' and software engineering principles.

SBQL queries can be embedded within statements that can change the database or program state. We follow the state-of-the-art known from majority of programming languages. Typical imperative constructs are creating a new object, deleting an object, assigning new value to an object (updating) and inserting an object into another object. We also introduce typical control and loop statements such as if…then…else…, while loops, for and for each iterators, and others. Some peculiarities are implied by queries that may return collections; thus, there are possibilities to generalize imperative constructs according to this new feature.

SBQL in ODRA project introduces also procedures, functions and methods. All procedural abstractions of SBQL can be invoked from any procedural abstractions with no limitations and can be recursive. SBQL programming abstractions deal with parameters being any queries; thus, corresponding parameter passing methods are generalized to take collections into account.

SBQL is a strongly typed language. Each database and program entity has to be associated with a type. However, types do not constraint semi-structured nature of the data. In particular, types allow for optional elements (similar to null values known from relational systems, but with different semantics – e.g., see Listing 1) and collections with arbitrary cardinality constraints. Strong typing of SBQL is a prerequisite for developing powerful query optimization methods based on query rewriting and on indices.

Listing 1. Sample SBQL query (Get employees who have salary and earn the same as Brown). Source: [30]

```
(Emp with salary) where salary =
              ((Emp with salary where name =
                              ”Brown”).salary);
```

C. *Virtual Updatable Views*

Another interesting and quite unique ODRA property are updatable views. Classical SQL views do the mapping from stored data into virtual data. However, some applications may require updating of virtual data; hence, there is a need for a reverse mapping: updates of virtual data are to be mapped into updates of stored data. This leads to the well-known view updating problem: updates of virtual data can be accomplished by updating of stored data on many ways, but the system cannot decide, which of them is to be chosen. In typical solutions these updates are made by side effects of view invocations. Due to the view updating problem, many kinds of view updates are limited or forbidden.

In the ODRA project (basing on previous research) another point of view has been introduced. In general, the method is based on overloading generic updating operations (create, delete, update, insert, etc.) acting on virtual objects by invocation of procedures that are written by the view definer (Listing 2). The procedures are an inherent part of the view definition. They have full algorithmic power, thus there are no limitations concerning the mapping of view updates into updates of stored data. SBQL updatable views allow one to achieve full transparency of virtual objects: they cannot be distinguished from stored objects by any programming option. This feature is very important for distributed and heterogeneous databases.

Listing 2. An SBQL updatable view definition. Source: [30]

```
view RichEmpDef {
 virtual RichEmp : record
                    {name:string;
                     salary:integer;
                     worksIn: ref Dept;}[0..*];
    seed: record {e: ref Emp;}[0..*] {
        return (Emp where salary > threshold) as e;
    }

    on_retrieve { return e.(name as name,
                            salary as salary,
                            ref (Dept where name =
                               deptName) as worksIn);
    }

    on_update {
                  e.name := value.name;
                  e.deptName := value.worksIn.name;
                  if(e.salary < value.salary) {
                    e.salary := value.salary;
                  }
    }

    on_new newEmp {
         if(newEmp.salary > threshold)
         create permanent Emp(
                newEmp.name as name,
                newEmp.salary as salary,
                newEmp.worksIn.name as deptName);
    }

    threshold: integer;
}
```

IV. OUR PROPOSAL

We believe that every developer should focus on the main task, e.g., creating a new application, a component or just a single function. In order to do this, his/her production environment should be as helpful as possible. This goal is fulfilled by modern IDEs. Of course, sometimes there are programmers who like to write their source code in a simple editor, but they are a minority. Most efficient IT professional use dedicated tools for their jobs. Thus, we have also decided to support programmers working with our WebODRA2 framework (and the ODRA DBMS in general) by introducing the ODRA eIDE2. The software together with tutorials is freely available; see [32].

A. ODRA eIDE2

There are different approaches to developing an IDE. In particular, there are the following possibilities:

- Start from scratch and manually create all necessary components;

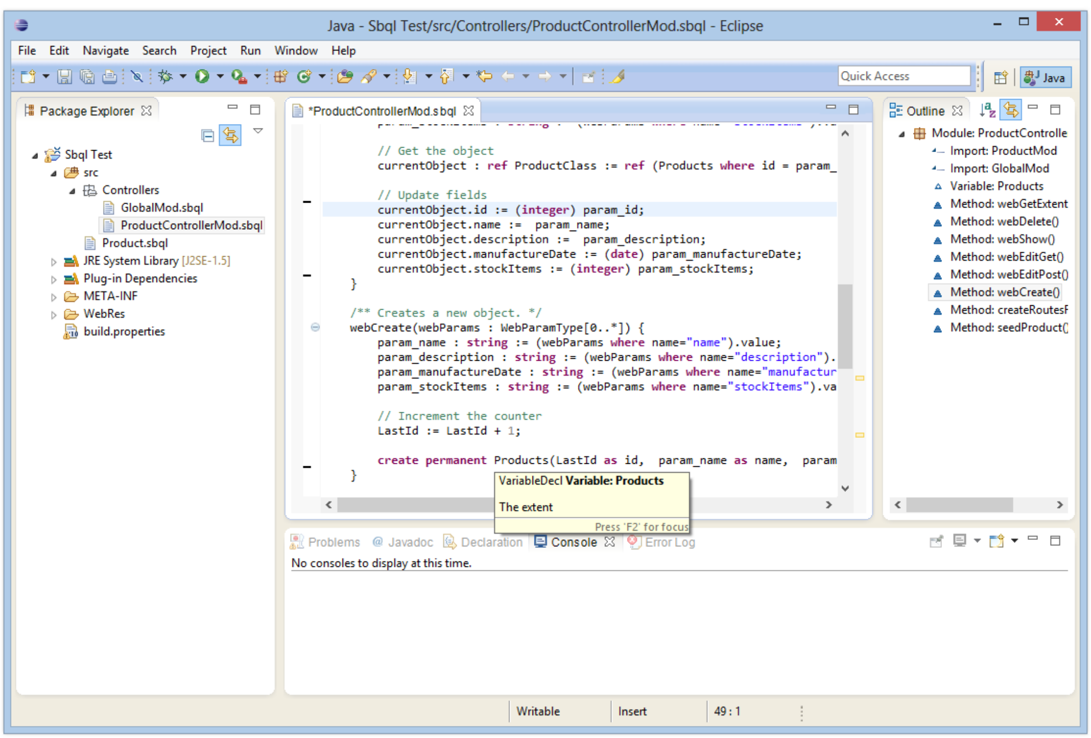


Figure 4. A window of ODRA eIDE2. Source: own elaboration.

- Extend an existing text editor. This approach was chosen by us a few years ago when we created the first ODRA IDE [33] based on the jEdit [34]. jEdit is a decent programmer's editor but does not provide the most advanced features of modern IDEs (see Section II.B). Thus, the IDE had syntax coloring, project management, error information, but lacked auto completion, refactoring, semantic navigation, etc.;
- Tailor an existing IDE to our needs. Depending on the IDE's architecture the process may require modification of the core code (which is not always publicly available) or create custom plugins.

Basically, modern IDEs are very complicated and sophisticated software. For instance, according to [35] the entire Juno edition of Eclipse (4.2.x) consists of 72 projects and contains 55 million lines of code. Obviously this is huge project and creating something similar requires a lot of resources.

After analyzing our expectations, which involved all previously described functionalities, we came to the conclusions that only the third approach is feasible for us. Therefore, we conducted some research to select the right IDE to extend.

Our choice was Eclipse together with Xtext framework [36]. The framework simplifies the process of creating our own DSL language together with a dedicated editor. From our point of view the latter feature was especially interesting.

1) The SBQL Grammar

In order to implement the functionalities like autocomplete or refactoring, the editor has to "understand" the source code. All particular artefacts of the supported language (e.g., class definitions or method parameters) have to be precisely defined and recognized in the source code (in the text). Usually, it could be achieved using a dedicated grammar describing the language. This is also the case of the framework, but there were some problems. The Xtext uses ANTLR parser, which utilizes LL(*) algorithm. Despite many advantages, the algorithm does not permit left recursion in grammars. On contrary, our existing ODRA compiler was based on Cup Parser Generator, which employs LALR(1) algorithm. Thus, we had to rewrite it taking into account the fundamental differences. The current version, implemented in the eIDE2, covers about 95% of the SBQL. Due to its size (about 400 lines of code and 80 production rules) we are not able to include it in the paper.

The grammar is just a starting point for more advanced functionalities. They have to be implemented by adding dedicated classes and/or methods. The more detailed description could be found in the next sections.

2) Basic Editor Functionalities

After defining the grammar, many features are available without any additional modifications (Figure 4). That includes things like: syntax coloring, text editing, basic navigation, tooltips (javadoc-like).

3) Content Assist

The content assist (autocomplete) is one of the most useful features provided by IDE (Figure 5). The Xtext framework contains a default implementation for this functionality. It could be modified by overriding some methods and/or adding new classes. We had to apply such modifications in some cases:

- The `import` statement. In the current version of the SBQL, the import statement adds entire module rather than a class (like in Java);
- Variable's declaration;
- The `where` expression should link to the content of the "inner" expression;
- The dot expression should be distinguished from the above mentioned `where`;

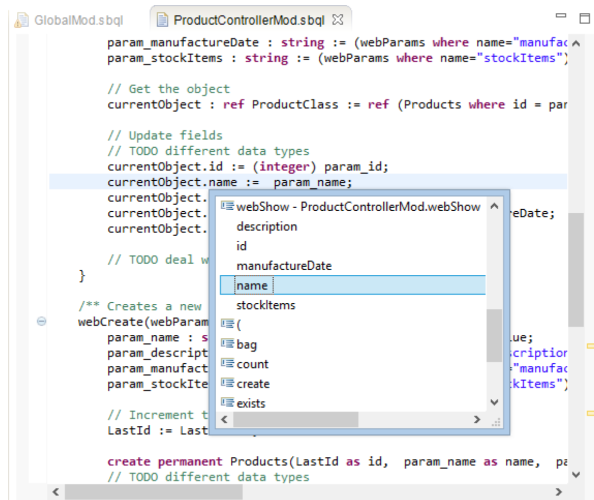


Figure 5. Autocomplete feature in eIDE2. Source: own elaboration.

4) Quick fixes

When IDE detects a problem related to edited code, it is marked as a warning. Some of the warnings could be automatically fixed. Currently, we have implemented a few quick fixes, e.g., for missing fields (Figure 6).

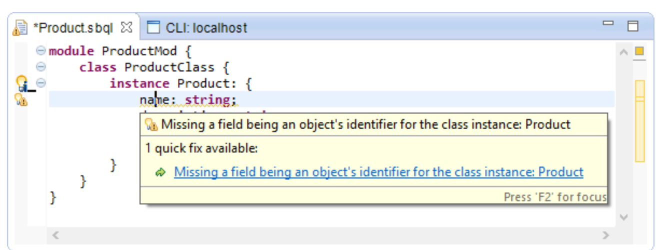


Figure 6. A quick fix in eIDE2. Source: own elaboration.

5) Outline Window

The outline window (Figure 7) shows a content's summary of the entire file. We modified the default implementation by adding custom graphical artefacts and altered some behaviors, e.g., presenting fields in the `Record` type.

6) Refactoring

The refactoring is probably the second most widely used feature (after the content assist). Currently, we have only the default implementation, which allows for renaming (Figure 8;

please note the active borders around the renamed element: `MyRoutes`).

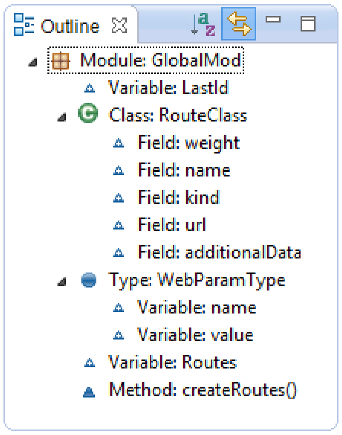


Figure 7. The outline window showing a semantic summary of the edited source file. Source: own elaboration.

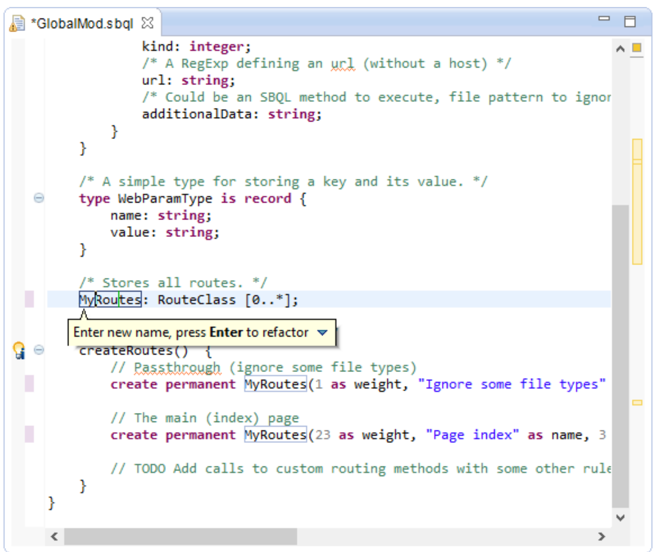


Figure 8. Refactoring in eIDE2. Source: own elaboration.

7) ODRA DBMS Integration

The ultimate goal of creating eIDE2 was facilitation of the ODRA development process both for its web framework and pure DBMS. Thus, we had to take care of easy integration. To do this, we introduced the following functionalities:

- Connection dialog (Figure 9). Every project managed by IDE could have its own, connected ODRA instance. It is possible to use just default values or provide custom ones.
- A programmer can also send text commands using a dedicated dialog window via CLI (Figure 10). This way of working with the DBMS allows for perform many advanced actions, currently not available directly from GUI.

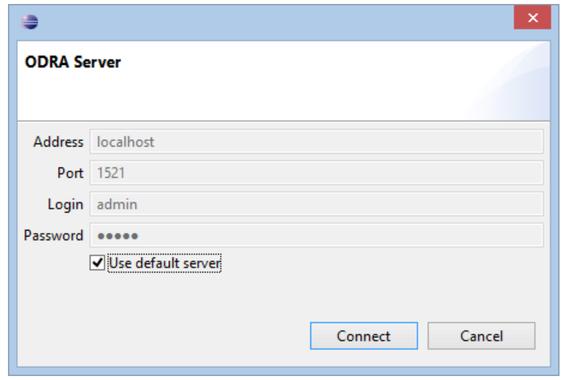


Figure 9. ODRA DBMS connection dialog (managed by eIDE2). Source: own elaboration.

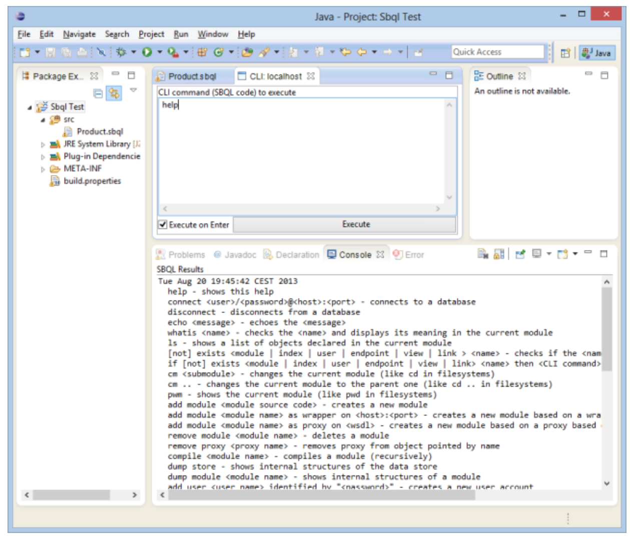


Figure 10. An access to ODRA DBMS CLI (Command Line Interface). Source: own elaboration.

- Executing a SBQL method. When the IDE detects a parameterless method a small icon is placed near the code. A user is able to execute the method directly from IDE on the connected ODRA DBMS (Figure 11).

8) The Scaffolding

In case of eIDE2, the scaffolding supports a programmer in creating a web GUI for WebODRA2. The generated GUI together with SBQL controllers add CRUD operations (Create, Retrieve, Update, Delete) using a client-side library called Knockout.js [37]. The functionality is easy-to-use yet powerful. For more information see Section V.

9) Other goodies

It is also worth noting that thanks to the Eclipse ecosystem it is possible to utilize many other 3-rd party plugins. They can provide additional possibilities including web editors, versioning systems (Subversion, Git), spellcheckers, UML tools, GUI editors.

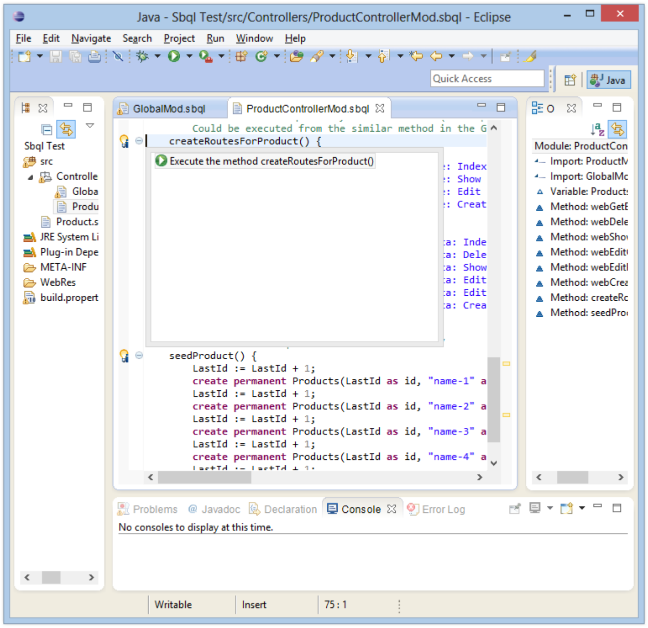


Figure 11. Executing an SBQL method in eIDE2. Source: own elaboration.

B. *WebODRA2*

Basically, every web application, no matter how it is developed, requires the following set of logical components:

- A graphical user interface,
- A routing system,
- A business logic,
- Data to work with.

The above components could be implemented using various approaches. In some cases, a programmer has to manually define them whereas other solutions use generators to create some of them automatically. Additionally, real world websites also require some static files: html templates, css, jpeg, etc.

We have decided to use pure programmatic approach, which means that all necessary definitions are provided by a programmer. It may look like a lot of work, but thanks to the high level of abstraction, the amount of information is significantly reduced.

Another feature, which simplifies development is the MVC (Model – View - Controller) architecture utilized in many previously mentioned frameworks. Comparing to the other frameworks, our approach uses the same object-oriented model both for a business logic (Controller) and data (Model). This method not only removes the impedance mismatch but also allows for using a powerful query and programming language for developing a business logic (behavior of the application). Furthermore, it is known that query languages operate on higher level of abstraction, effectively reducing the amount of code that needs to be written to achieve the same goals. For instance, a few tenths lines of Java code could be equivalent to a literally few lines of SBQL (or SQL). Not to mention performance and various optimizations, which are much more advanced in query languages.

Another very important area of a web framework is a graphical user interface. There are different methods to deal with the topic, some of them follow the MVC pattern. One of the most popular is using a server-side templating engine. A template contains an HTML code mixed with special tags, usually provided by the framework. In most cases, the tags allow to embed parts of a programming language (e.g., Java), mainly to insert some data (e.g., a list of products or customers). However, some programmers use them to implement additional functionality, which duplicates the controller's responsibility. Of course, it is an incorrect application of the tags affecting maintainability of the code. At the end, tags are processed by an engine, a final HTML page is generated and sent to a web browser.

Comparing to the first release (see [1]), the second edition (WebODRA2) contains some bug fixes and modifications related to configuration and a project's structure. They were mainly caused by the support in the eIDE2.

Figure 12 contains a simplified logical architecture of our prototype framework for developing web application called WebODRA2. The framework consist of two principal parts:

- A web server. It is responsible for responding to incoming requests from a web browser. The implementation of the server is based on open source tool called Jetty [38];
- ODRA Database Management System. This is a standard instance of the ODRA server introduced in Section 3.

The following subsections describe each of the components (from Figure 12) in details.

1) *Routing Module*

The center of WebODRA2 consists of a routing module, which is responsible for a correct processing of incoming web requests. The module is driven by rules defined by a programmer. Each definition, written in SBQL (as an object with specific properties), contains the following information:

- Url. A regular expression, which will be applied to the incoming request's url. If there is a match, then the rule will be executed;
- Weight. It affects an order of the processing;
- Name. Human-readable name of the rule. It is especially useful during logging;
- Additional Data. The utilization of the additional data depends on a rule kind;
- Rule Kind. It affects the following processing:
  - Passthrough. The web framework ignores those rules and they are processed by the Jetty server. They serve static files like: pictures, css, Java script, etc.;
  - Data route. They contain an SBQL method's name to execute. The method will get all HTML form parameters entered by a user, which makes it possible to process them by an SBQL code. The result of the method is transformed (see further) and returned to the browser;

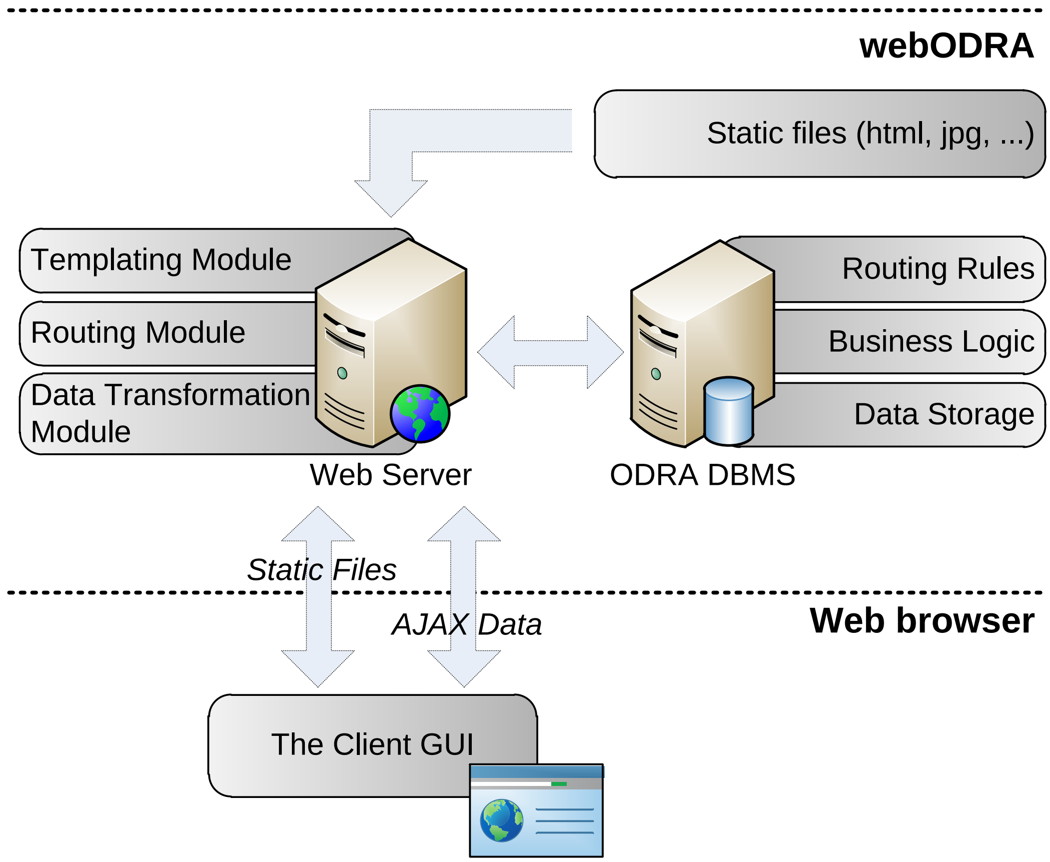


Figure 12. Logical architecture of WebODRA2. Source: own elaboration.

- Page route. An HTML page post-processed by our simple templating engine (see further).

2) *The Client GUI*

As previously mentioned, typical server-side web templating engines may lead to overuse tags by implementing some business functionality. To prevent this we have decided to use a client-side GUI framework. The idea is based on embedding in a web page some (meta) information, which will be used to present business data. We have chosen a framework called Knockout [37] utilizing new HTML5 *data-* attributes. They allow to create custom attributes and store any information. The process of showing a web page contains two steps. First, an HTML page is downloaded from a server, containing the markers. Then, the library sends an AJAX request to asynchronously retrieve necessary data, which are "injected" into the page.

The user data submission is performed on a similar rules. An asynchronous request is send to the server, triggering a Data Rule processing the provided data.

Standard website navigation is performed using regular hyperlinks ("outside" the framework).

We do not provide any dedicated GUI controls as a part of the framework. However, some of them are generated during the scaffolding process (see Section V). A programmer is also free to use any available solutions as long as they could be tuned to work with JSON format (e.g., some JavaScript libraries/frameworks).

3) *Templating Module*

The templating module is responsible for a coherent look and fill of the entire website. It operates on a single master page, which has a dynamic area fulfilled with some functional pages, i.e., a document repository, a forum, news, etc. For instance, the master page can contain a header, a navigation panel and a footer.

The process is triggered by the Page Route rule. When a particular page is requested by a browser, the master page is applied, or to be more precise, the requested page is embedded in the master page and then returned to the browser.

4) *Data Transformation Module*

When a Data Route rule executes a given SBQL method, the result could be any SBQL data type, e.g., a collection, a single object or a text. It needs to be processed to the format recognized by the Client GUI. The Data Transformation Module recursively converts the result into JSON [7] string, sends it back to the web browser where it is further processed.

5) *Routing Rules, Business Logic and Data Storage*

The above components are described in other Sections:

- The Routing Rules in Section IV.B.1;
- The Business Logic stores appropriate SBQL code (see Section III.B) referenced from *Data routes*;

- The Data Storage uses ODRA DBMS (see Section III).

V. SCAFFOLDING FOR THE WEBODRA2

As previously mentioned, scaffolding is a major programmer's facilitation in creating real-world web applications. It generates (one-way) necessary artefacts (e.g., source code, media and HTML files) being a starting point for further development.

In case of eIDE2, for each SBQL class a dedicated button is placed near its source definition. Then, it is possible to start the process and all necessary files and folders will be generated (Figure 13):

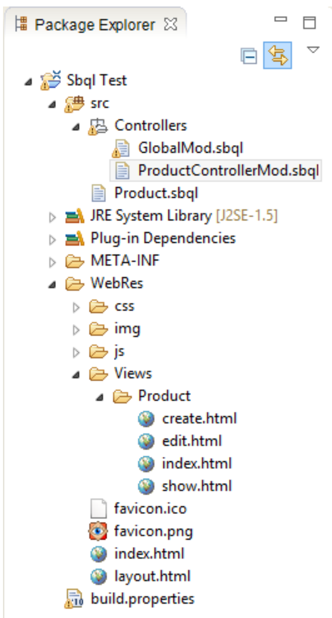


Figure 13. Files scaffolded by eIDE2. Source: own elaboration.

- *GlobalMod.sbql*. This file stores some global data, common for the entire project:
  - Two definitions: `RouteClass` (each instance stores a single routing rule) and `WebParamType` (utilized for passing parameters from the web);
  - `Routes` acting as an extent for all the rules;
  - A `createRoutes()` method, which puts global routing rules into the DB;
- *ProductControllerMod.sbql*. This is a controller for the business class `Product`. It is responsible for:
  - Storing all instances of `Product`;
  - Processing web requests (all methods: `webXXX`. The methods are referenced by *Data routing* rules (inside the `createRoutesForProduct()` method));
  - Creating routes for CRUD operations for the Product class (the `createRoutesForProduct()` method);
  - Generating sample data (the `seedProduct()` method);
- *WebRes* folder, which is a root folder for the web server utilized by the framework. Aside of self-explaining typical folders (*css*, *img*, *js*) there are some others worth a short discussion:
  - The *Views* folder stores subdirectories for each business class. The subdirectories contain dedicated *html* files for processing particular CRUD operations. The files are referenced by *Page routing* rules (inside the `createRoutesForProduct()` method);
  - *index.html* file is a starting point for a web navigation;
  - *layout.html* file is a master template file providing a coherent look and feel for the entire site;

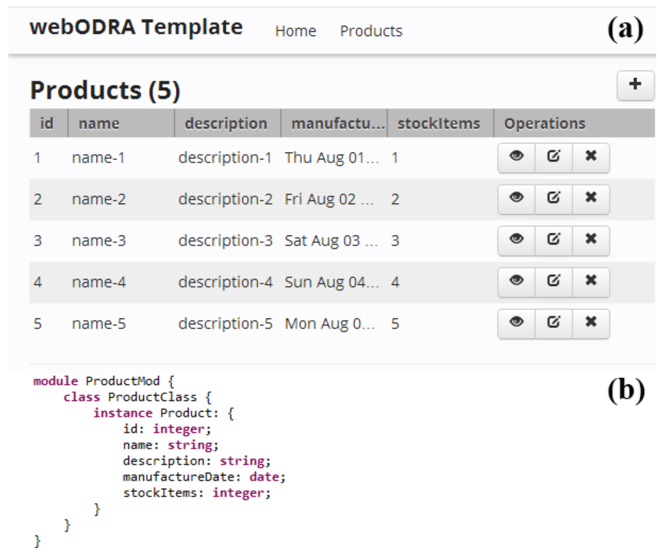


Figure 14. One of the views (index) generated by the scaffolding (a) for the sample Product class (b). Source: own elaboration.

To sum up, a programmer using just one click can create a simple prototype of a working web application (Figure 14). The generated text files contains the code (both SBQL and HTML), which could be easily edited using the IDE or any external tools.

Currently, WebODRA2 is shipped with two examples: the forum (described in [1]) and the scaffolded one described above.

VI. CONCLUSION AND FUTURE WORK

We have presented our approach to creating web applications using a single, coherent model utilized both for data and business logic. Thanks to the powerful query and programming language SBQL, a programmer stays on the same high level of abstraction, saving time and making less errors.

Our approach is supported by two tools working together: a web framework called WebODRA2 and a dedicated IDE (eIDE2). Furthermore, we have added a scaffolding mechanism generating a web GUI with CRUD operations for any business class. The software is freely available online together with some tutorials [32].

The contribution of this paper is based on quite new method for creating websites. To our best effort, we were not able to find a similar solution, directly employing the power of a modern database to develop web portals.

We believe that this kind of solutions could be a valuable alternative to existing tools for creating data intensive web applications. Thus, we would like to continue our research in that field, improving our framework and the eIDE2 to make them production-ready.

REFERENCES

[1] M. Trzaska, "WebODRA - A Web Framework for the Object-Oriented DBMS ODRA," in *The First International Conference on Building and Exploring Web Based Environments (WEB 2013)*, Seville, Spain, IARIA XPS Press, 2013, pp. 1-6.

[2] M. Trzaska, "Data Migration and Validation Using the Smart Persistence Layer 2.0," in *The 16th IASTED International Conference on Software Engineering and Applications (SEA 2012)*, Las Vegas, USA, Acta Press, November 12 – 14, 2012, pp. 186-193.

[3] M. Perscheid, D. Tibbe, M. Beck, S. Berger, P. Osburg, J. Eastman, M. Haupt and R. Hirschfeld, An Introduction to Seaside, Hasso-Plattner-Institut, 2008.

[4] "ActiveRecord," [Online]. Available: http://api.rubyonrails.org/classes/ActiveRecord/Base.html [Accessed 05.09.2013].

[5] "CouchApp," [Online]. Available: http://couchapp.org/page/index. [Accessed 21.08.2012].

[6] C. Anderson, J. Lehnardt and N. Slater, CouchDB: The Definitive Guide, O'Reilly Media, 2010.

[7] "JSON: JavaScript Object Notation," [Online]. Available: http://www.json.org/. [Accessed 01.09.2013].

[8] D. Distante, P. Pedone, G. Rossi and G. Canfora, "Model-Driven Development of Web Ap-plications with UWA, MVC and JavaServer Faces," in *Web Engineering Lecture Notes in Computer Science*, Springer, 2007, pp. 457-472.

[9] Oracle, "Oracle Application Developer's Guide," [Online]. Available: http://docs.oracle.com/. [Accessed 05.09.2013].

[10] D. Arraes Nunes and D. Schwabe, "Rapid Prototyping of Web Applications combining Do-main Specific Languages and Model Driven Design," in *Proceedings of the 6th International Conference on Web Engineering (ICWE'06)*, Palo Alto, California, USA, July 11-14, 2006.

[11] S. Ceri, P. Fraternali and M. Matera, "Conceptual Modeling of Data-Intensive Web Applications," *IEEE Internet Computing vol. 6, no. 4,* July/August 2002.

[12] J. Williamson, Oracle Application Express: Fast Track to Modern Web Applications, McGraw-Hill Osborne Media, 2012.

[13] "Eclipse," [Online]. Available: http://www.eclipse.org/. [Accessed 6.9.2013].

[14] "NetBeans IDE," [Online]. Available: https://netbeans.org/. [Accessed 06.09.2013].

[15] "IntelliJ IDEA," [Online]. Available: http://www.jetbrains.com/idea/. [Accessed 04.09.2013].

[16] "Visual Studio," [Online]. Available: http://www.microsoft.com/visualstudio/plk. [Accessed 03.09.2013].

[17] "MonoDevelop," [Online]. Available: http://monodevelop.com/. [Accessed 05.09.2013].

[18] "SharpDevelop," [Online]. Available: http://www.icsharpcode.net/opensource/sd/. [Accessed 01.09.2013].

[19] "Aptana Studio," [Online]. Available: http://www.aptana.com/. [Accessed 02.09.2013].

[20] "Eclipse PHP Development Tools," [Online]. Available: http://projects.eclipse.org/projects/tools.pdt. [Accessed 04.09.2013].

[21] "PhpStorm," [Online]. Available: http://www.jetbrains.com/phpstorm/. [Accessed 02.09.2013].

[22] "KDevelop," [Online]. Available: http://www.kdevelop.org/. [Accessed 02.09.2013].

[23] "Pharo," [Online]. Available: http://www.pharo-project.org/home. [Accessed 02.09.2013].

[24] "VisualWorks," [Online]. Available: http://www.cincomsmalltalk.com/main/products/visualworks/. [Accessed 04.09.2013].

[25] "RubyMine," [Online]. Available: http://www.jetbrains.com/ruby/. [Accessed 03.09.2013].

[26] "Ruby and Rails - plugin for NB," [Online]. Available: http://plugins.netbeans.org/plugin/38549. [Accessed 04.09.2013].

[27] "Aptana RadRails," [Online]. Available: http://www.aptana.com/products/radrails. [Accessed 04.09.2013].

[28] K. Subieta, "Stack-based Query Language," in *Encyclopedia of Database Systems*, Springer US, 2009, pp. 2771-2772.

[29] R. Adamus, P. Habela, K. Kaczmarski, M. Lentner, T. Pieciukiewicz, K. Stencel, K. Subieta, M. Trzaska and J. Wislicki, "Overview of the Project ODRA," in *Proceedings of First International Conference on Object Databases (ICOODB) 2008*, Berlin, Germany, 2008.

[30] K. Subieta, "Stack-Based Architecture (SBA) and Stack-Based Query Language (SBQL)," [Online]. Available: http://www.sbql.pl/. [Accessed 01.09.2013].

[31] "ODRA (Object Database for Rapid Application development): Description and programmer manual," [Online]. Available: http://www.sbql.pl/various/ODRA/ODRA_manual.html. [Accessed 02.09.2013].

[32] "ODRA eIDE2 Home," [Online]. Available: http://www.eide2.pjwstk.edu.pl/. [Accessed 07.09.2013].

[33] "ODRA IDE (jEdit)," [Online]. Available: http://www.mtrzaska.com/odra. [Accessed 04.09.2013].

[34] "jEdit - Porgammer's Text Editor," [Online]. Available: http://www.jedit.org/.

[35] "Eclipse Juno Release Train Has Arrived," 27.06.2012. [Online]. Available: http://www.eclipse.org/org/press-release/20120627_junorelease.php. [Accessed 06.09.2013].

[36] "Xtext - Language Development Made Easy," [Online]. Available: http://www.eclipse.org/Xtext/. [Accessed 01.09.2013].

[37] "Knockout Framework," [Online]. Available: http://knockoutjs.com/. [Accessed 16.08.2013].

[38] "Jetty - Web Server," [Online]. Available: http://jetty.codehaus.org/jetty/. [Accessed 18.08.2012].

[39] "The WebODRA Framework," [Online]. Available: http://www.mtrzaska.com/webodra. [Accessed 26.08.2013].

On How to Provision Virtual Circuits for Network-Redirected Large-Sized, High-Rate Flows

Zhenzhen Yan, Malathi Veeraraghavan
Dept. of Electrical and Computer Engineering
University of Virginia
Charlottesville, VA, USA
Email: {zy4d,mvee}@virginia.edu

Chris Tracy, Chin Guok
Energy Sciences Network (ESnet)
LBNL
Berkeley, CA, USA
Email: {ctracy,chin}@es.net

***Abstract*—To reduce the impact of large-sized, high-rate (α) transfers on real-time flows, a Hybrid Network Traffic Engineering System (HNTES) was proposed in earlier work. HNTES is an intra-domain solution that enables the automatic identification of α flows at a provider network's ingress routers, and redirects these flows to traffic-engineered QoS-controlled virtual circuits. The purpose of this work is to determine the best QoS mechanisms for the virtual circuits used in this application. Our findings are that a no-policing, two-queues solution with weighted fair queueing and priority queueing is both sufficient and the best for this application. It allows for the dual goals of reduced delay/jitter in real-time flows, and high-throughput for the α flows, to be met.**

Keywords*—*policing; scheduling; high-speed networks; traffic-engineering; virtual-circuit networks

I. Introduction

This paper is an extended version of a published conference paper [1]. It describes a set of experiments that were conducted to determine the best Quality-of-Service (QoS) mechanisms to apply while redirecting large-sized high-rate flows to virtual circuits within provider networks.

To move large datasets, scientists typically invest in high-end computing systems that can source and sink data to/from their disk systems at high speeds. These transfers are referred to as α flows as they dominate other flows [2]. They also cause increased burstiness, which in turn impacts delay-sensitive real-time audio/video flows. In prior work [3], we proposed an overall architecture for an intra-domain traffic engineering system called Hybrid Network Traffic Engineering System (HNTES) that performs two tasks: (i) analyzes NetFlow reports offline to identify α flows, and (ii) configures the ingress routers for future α-flow redirection to traffic-engineered Quality-of-Service (QoS)-controlled paths. The prior paper [3] then focused on the first aspect, and analyzed NetFlow data obtained from live ESnet routers for the period May to Nov. 2011. The analysis showed that since α flows require high-end computing systems to source/sink data at high speeds, these systems are typically assigned static global public IP addresses, and repeated α flows are observed between the same pairs of hosts. Therefore, source and destination address prefixes of observed α flows can be used to configure firewall filter rules at ingress routers for future α-flow redirection. The effectiveness of such an offline α-flow identification scheme was evaluated with the collected NetFlow data and found to be 94%, i.e., a majority of bytes sent in bursts by α flows would have been successfully isolated had such a traffic engineering system been deployed [3].

The work presented here focuses on the second aspect of HNTES by addressing the question of how to achieve α-flow redirection and isolation to traffic-engineered paths. Specifically, service providers such as ESnet [4] are interested in actively selecting traffic-engineered paths for α-flows, and using QoS mechanisms to isolate these flows. With virtual-circuit technologies, such as MultiProtocol Label Switching (MPLS), ESnet and other research and education network providers, such as Internet2, GEANT [5], and JGN-X [6], offer a dynamic circuit service. An On-Demand Secure Circuits and Advance Reservation System (OSCARS) Inter-Domain Controller (IDC) [7] is used for circuit scheduling and provisioning.

The basic interface to the IDC requires an application to specify the circuit rate, duration, start time, and the endpoints in its advance-reservation request. The specified rate is used both for (i) path computation in the call-admission/circuit-scheduling phase and (ii) policing traffic in the data plane. If the application requests a high rate for the circuit, the request could be rejected by the OSCARS IDC due to a lack of resources. On the other hand, if the request is for a relatively low rate (such as 1 Gbps), then the policing mechanism could become a limiting factor to the throughput of α flows, preventing TCP from increasing its sending rate.

The purpose of this paper is to evaluate the effects of different scheduling and policing mechanisms to achieve two goals: (i) reduce delay and jitter of real-time sensitive flows that share the same interfaces as α flows, and (ii) achieve high throughput for α-flow transfers.

Our *key findings* are as follows: (i) With the current widely deployed best-effort IP-routed service, which uses first-come-first-serve (FCFS) packet scheduling on egress interfaces of routers, the presence of an α flow can increase the delay and jitter experienced by audio/video flows. (ii) This influence can be eliminated by configuring two virtual queues at the contending interface and redirecting identified α flows to one queue (α queue), while all other flows are directed to a second queue (β queue). (iii) The policer should not be configured to direct out-of-profile packets of an α TCP flow to a different queue from its in-profile packets. When packets of the same TCP flow are served from different queues, packets can arrive out of sequence at the receiver. Out-of-sequence arrivals triggers TCP's fast retransmit/fast recovery congestion algorithm, which causes the TCP sender to lower its sending rate resulting in degraded throughput. (iv) An alternative approach to dealing with out-of-profile packets is to probabilistically drop a few packets using Weighted Random Early Detection (WRED), and to buffer the remaining out-of-profile packets in the same queue as the in-profile packets. This prevents the out-of-sequence problem and results in a smaller drop in α-flow throughput when compared to the separate-queues approach. (v) The no-policing scheme is preferred to the policing/WRED scheme because HNTES redirects α flows within a provider's network, which means that these flows will typically run TCP and are not rate-limited to the circuit rate. If an end application requested a circuit explicitly, then it can be expected to use traffic control mechanisms, such as Linux `tc`, to limit the sending rate. But with HNTES, the end application is not involved in the circuit setup phase, and therefore the applications are likely to be running unfettered TCP. Under these conditions, when buffer occupancy builds up, packets will be deliberately dropped in the policing/WRED scheme, leading to poor performance. Furthermore, if there are two simultaneous α flows, the probability of buffer buildups increases, which in turn increases the dropped-packet rate and lowers throughput. This recommendation of using a no-policing only scheme for α flows does not prevent the application of other QoS mechanisms to real-time flows after they have been separated out from α flows. (vi) The negatives of partitioning rate/buffer space resources between two queues were studied. Our conclusions are that close network monitoring is required to dynamically adjust the rate/buffer space split between the two queues as traffic changes, and the probability of unidentified α flows should be reduced whenever possible to avoid these flows from becoming directed to the β queue.

Section II provides background and reviews related work. Section III describes the experiments we conducted on a high-speed testbed to evaluate different combinations of QoS mechanisms and parameter values to achieve our dual goals of reduced delay/jitter for real-time flows and high throughput for α flows. Our conclusions are presented in Section IV.

II. Background and related work

The first three topics, historical perspective, a hybrid network traffic engineering system, and QoS support in state-of-the-art routers, provide the reader with relevant background information. The last topic, QoS mechanisms applied to TCP flows, covers related work.

Historical perspective: In the nineties, when Asynchronous Transfer Mode (ATM) [8] and Integrated Services (IntServ) [9] technologies were developed, virtual circuit (VC) services were considered for delay-sensitive multimedia flows. However, these solutions are not scalable to large numbers of flows because of the challenges in implementing QoS mechanisms such as policing and scheduling on a per-flow basis. Instead, a solution of overprovisioning connectionless IP networks has been affordable so far. Overprovisioning prevents router-buffer buildups and thus ensures low delay/jitter for real-time audio/video flows. While this solution works well most of the time, there are occasional periods when a single large dataset transfer is able to ramp up to a very high rate and adversely affect other traffic [10]. Such transfers, which are referred to as α flows, occur when the amount of data being moved is large, and the end-to-end sustained rate is high.

In the last ten years, there has been an emergent interest in using VCs but for α-flow transfers not multimedia flows. As noted in Section I, service providers are interested in routing these α flows to traffic-engineered, QoS-controlled paths. The scalability issue is less of a problem here since the number of α flows is much smaller than of that of real-time audio-video flows. It is interesting to observe this "flip" in the type of applications being considered for virtual-circuit services, i.e., from real-time multimedia flows to file-transfer flows.

Hybrid Network Traffic Engineering System (HNTES): Ideally if end-user applications such as GridFTP [11] alerted the provider networks en route between the source and destination before starting a high-rate, large-sized dataset transfer, these networks could perform path-selection and direct the resulting TCP flow(s) to traffic-engineered, QoS-controlled paths. However, most end-user applications do not have this capability, and furthermore inter-domain signaling to establish such paths requires significant standardization efforts. Meanwhile, providers have recognized that intra-domain traffic-engineering is sufficient if α flows can be automatically identified at the ingress routers. Deployment of such a traffic-engineering system lies within the control of individual provider networks, making it a more attractive solution. Therefore, the first step in our work was to determine whether such automatic α flow identification is feasible or not.

In our prior work [3], we started with a hypothesis that computers capable of sourcing/sinking data at high rates are typically allocated static public IP addresses, and α flows between pairs of these computers occur repeatedly as the same users initiate dataset transfers. This hypothesis was true for ESnet traffic. Therefore, HNTES can determine source-destination IP address prefixes by analyzing NetFlow reports of completed α flows and use these address prefixes to set firewall filters to redirect future α flows. Our heuristic was simple: if a NetFlow report for a flow showed that more than H bytes (set to 1 GB) were sent within a fixed time interval (set to 1 min), we classified the flow as an α flow. This NetFlow data analysis is envisioned to be carried out offline on say a nightly basis for all ingress routers to update the firewall filters. If no flows are observed for a particular source-destination address prefix within an aging interval (set to 30 days), then the firewall filter entry is removed. The effectiveness of this scheme was evaluated through an analysis of 7 months of NetFlow data obtained from an ESnet router. For this data set, 94% (82%) of bytes generated by α flows in bursts would have been identified correctly and isolated had /24 (/32) based prefix IDs been used in the firewall filters.

QoS support in routers: Multiple policing, scheduling and traffic shaping mechanisms have been implemented in today's routers. While new mechanisms such as Flow-Aware Networking (FAN) [12] are being developed, in this section, we review the particular mechanisms used in ESnet routers, and hence in our experiments. For scheduling, two mechanisms are used: Weighted Fair Queueing (WFQ) and Priority Queueing (PQ) [13]. With WFQ, multiple traffic classes are defined, and corresponding virtual queues are created on egress interfaces. Bandwidth can be strictly partitioned or shared among the virtual queues. WFQ is combined with PQ as explained later. On the ingress-side, policing is used to ensure that a flow does not exceed its assigned rate (set by the IDC during call admission). For example, in a single-rate two-color (token bucket) scheme, the average rate (which is the rate specified to the IDC in the circuit request) is set to equal the generation rate of tokens, and a maximum burst-size is used to limit the number of tokens in the bucket. The policer marks packets as *in-profile* or *out-of-profile*. Three different actions can be configured: (i) discard out-of-profile packets immediately, (ii) classify out-of-profile packets as belonging to a `Scavenger Service (SS)` class, and direct these packets to an `SS` virtual queue, or (iii) drop out-of-profile packets according to a WRED profile, but store remaining out-of-profile packets in the same queue as in-profile packets. For example, the drop rate for out-of-profile packets can be configured to increase linearly from 0 to 100 for corresponding levels of queue occupancy.

QoS mechanisms applied to TCP flows: Many QoS provisioning algorithms that involve some form of active queue management (AQM) have been studied [14]–[18]. Some of the simpler algorithms have been implemented in today's routers, such as RED [14] and WRED [16], while other algorithms, such as Approximate Fair Dropping (AFD) [18], have been shown to provide better fairness. An analysis of the configuration scripts used in core and edge routers of ESnet shows that these AQM related algorithms are not enabled. This is likely due to the commonly adopted policy of overprovisioning (an Internet2 memorandum [19] states a policy of operating links at 20% occupancy). Nevertheless, providers have recognized that in spite of the headroom, an occasional α flow can spike to a significant fraction of link capacity (e.g., our GridFTP log analysis showed average flow throughput of over 4 Gbps across 10-Gbps paths [10]). When the flow throughput averaged across its lifetime is 4 Gbps, there can be short intervals in which the flow rate spiked to values close to link capacity.

III. EXPERIMENTS

A set of experiments were designed and executed to determine the best combination of QoS mechanisms with corresponding parameter settings in order to achieve our dual goals of reduced delay/jitter for real-time traffic and high throughput for α flows. For the first goal, we formulated a hypothesis as follows: a scheduling-only no-policing scheme that isolates α-flow packets into a separate virtual queue is sufficient to keep non-α flow delay/jitter low. For the second goal, we experimented with different QoS mechanisms and parameter settings.

Experiment 1 was designed to understand the two modes for sharing link rate (strictly partitioned and work conversing), and to determine the router buffer size. *Experiment 2* tests the above-stated hypothesis for the first goal. *Experiments 3 and 4* studied two different mechanisms, using a separate scavenger-service (SS) queue vs. using WRED, for handling the out-of-profile packets identified by ingress-side policing, and compared results with a no-policing approach. We concluded that the WRED scheme was better, but it was outperformed by the no-policing scheme. *Experiment 5* was designed to check if the policing/WRED scheme had a fairness advantage over the no-policing scheme. We found that since neither of the two policed α flows honored their assigned rates (which should be expected for HNTES-redirected flows), under the no-policing scheme the TCP flows adjusted their sending rates and had no packet losses, while the deliberate packet losses introduced in the policing/WRED scheme lowered throughput for both flows, and furthermore resulted in lower fairness because of a difference in RTTs, even though this difference was small. In *Experiment 6*, we characterized the the impact of QoS provisioning under changing traffic conditions, and compared two versions of TCP: Reno and H-TCP. In the presence of an α flow that uses up its whole α-queue rate allocation, if the background traffic is more than the rate allocated to the β queue, the latter will suffer from more losses than if there had been no partitioning of resources between the two queues. This implies a need for closer monitoring of traffic and dynamic reconfiguration of the rate/buffer allocations to the two queues. However, since two rare events, an α flow and an increased background load, have to occur simultaneously, the probability of this scenario is low. H-TCP is better than Reno for high-speed transfers, but from the perspective of the impact on other flows, we did not see a significant difference in our tested scenarios. *Experiment 7* was designed to study the effects of an unidentified α flow being directed to the β queue. Here again, if there was no simultaneous α flow directed to the α queue when the unidentified α flow appeared, then the impact will be the same as without partitioning. However, if this combination of rare events occurs jointly, then given that the β queue has only a partition of the total interface rate/buffer space, the impact on delay-sensitive flows will be greater than if there had been no partitioning.

Section III-A describes the experimental setup, the experimental methodology, and certain router configurations that are common to all the experiments. The remaining subsections describe the seven experiments.

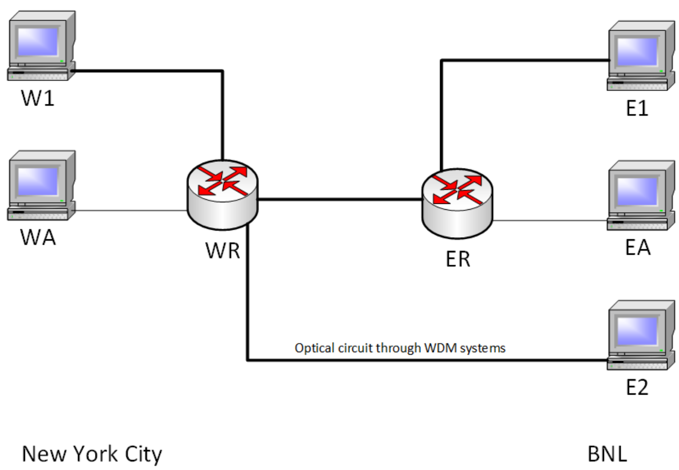


Figure 1. Experiment setup.

A. Experimental setup

The experimental network setup is shown in Figure 1. It was called the Long Island MAN (LIMAN) testbed, and was supported by ESnet as a DOE-funded testbed for networking research. The high-performance hosts, `W1` (West 1), `E1` (East 1), and `E2` (East 2), were Intel Xeon Nehalem E5530 models (2.4GHz CPU, 24GB memory) and ran Linux version 2.6.33. The application hosts, `WA` (West App-host) and `EA` (East App-host), were Intel Dual 2.5GHz Xeon model and ran Linux 2.6.18. The routers, `WR` (West Router) and `ER` (East Router), were Juniper MX80's running Junos version 10.2. The link rates were 10 Gbps from the high-performance hosts to the routers, 1 Gbps from the application hosts to the routers, and 10 Gbps between the routers.

Host `W1` and router `WR` were physically located in New York City, while the East-side hosts and routers, and host `E2`, were physically located in the Brookhaven National Laboratory (BNL) in Long Island, New York. Host `E2` was connected to router `WR` via a circuit provisioned across the Infinera systems of the underlying optical network as shown in Figure 1.

Each experiment consists of four steps: (i) plan the applications required to test a particular QoS mechanism, (ii) configure routers to execute the selected QoS mechanisms with corresponding parameter settings based on the planned application flows, (iii) execute applications on end hosts to create different types of flows through the routers, and (iv) obtain measurements for various characteristics, e.g., throughput, packet loss, and delay, from the end-host applications as well as from packet counters in the routers.

A preliminary set of experiments were conducted to determine the specific manner in which the egress-side link capacity was shared among multiple virtual queues. Theoretically, the transmitter can be strictly partitioned or shared in a work-conserving manner. If strictly partitioned, then even if there are no packets waiting in one virtual queue, the transmitter will not serve packets waiting in another queue. In this mode, each queue is served at the exact fractional rate assigned to it. In contrast, in the work-conserving mode the transmitter will serve additional packets from a virtual queue that is experiencing a higher arrival rate than its assigned rate if there are no packets to serve from the other virtual queues. The buffer is always strictly partitioned between the virtual queues in the routers used in our experiments.

Figure 2 illustrates how a combination of QoS mechanisms was used in our experiments. *First*, incoming packets are classified into multiple classes based on pre-configured firewall filters, e.g., α-flow packets are identified by the source-destination IP address prefixes and classified into the α class. *Second*, packets in some of these classes are directly sent to corresponding egress-side virtual queues, while flows corresponding to other classes are subject to policing. A single-rate token bucket scheme is applied. If an arriving packet finds a token in the bucket, it is marked as being in-profile; otherwise, it is marked as being out-of-profile. *Third*, for some policed flows, in-profile and out-of-profile packets are sent to separate egress-side virtual queues, while packets from other policed flows are subject to WRED before being buffered in a single virtual queue. On the egress-side, each virtual queue is assigned a priority level, a fractional allocation (expressed as a percentage) of link capacity, and a fractional allocation of the

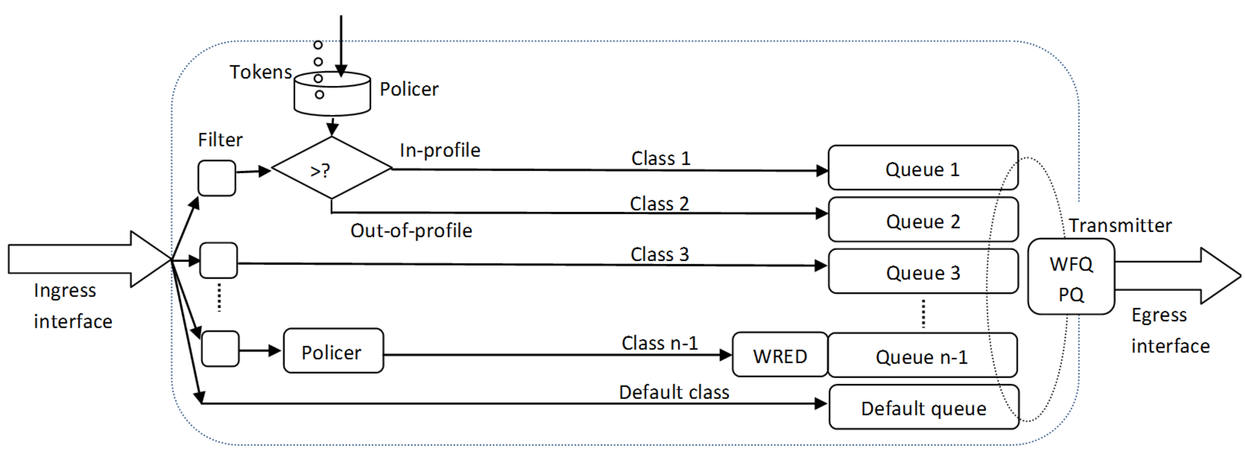


Figure 2. Illustration of QoS mechanisms in a router.

buffer. As noted in the previous paragraph the buffer allocation is strictly partitioned while the transmitter is shared in work-conserving mode. *Fourth*, the WFQ scheduler decides whether a virtual "queue is in-profile or not," by comparing the rate allocated to the queue and the rate at which packets have been served out of the queue. *Finally*, the PQ scheduler selects the queue from which to serve packets using their assigned priority levels, but to avoid starvation of low-priority queues, as soon as a large enough number of packets are served from a high-priority queue to cause the status of the queue to transition to out-of-profile, the PQ scheduler switches to the next queue in the priority ordering. When all queues become out-of-profile, it starts serving packets again in priority order. It is interesting that while the *policer* is flagging *packets* as in-profile or out-of-profile on a per-flow basis, the *WFQ scheduler* is marking *queues* as being in-profile or out-of-profile.

B. Experiment 1

1) Purpose and execution: The goals of this experiment were to (i) determine the router buffer size, (ii) determine the default mode used in the routers for link capacity (rate) sharing (between the two options of strict partitioning and work-conserving), and (iii) compare these two modes. Correspondingly, three scenarios were tested with different router configurations. To control rate and buffer allocations, the router software required the configuration of a virtual queue on the egress interface, even if it was just a single queue to which all flows were directed. In scenario 1, by modifying the buffer allocation for the virtual queue, router buffer size was determined. In scenario 2, by modifying the rate allocation, the default mode for capacity sharing was determined. Finally, in scenario 3, the router was explicitly configured to operate in the two different modes for comparison.

As per our execution methodology, the first step was to plan applications. For the first two scenarios, we planned to use two UDP flows created by the `nuttcp` application, and a "ping" flow to send repeated echo-request messages and receive responses. The purpose of the ping flow was to measure round-trip delays. While other applications could be used to emulate delay-sensitive flows, we chose a simple ping flow as it was sufficient for our needs. The UDP flow was used to fill up the router buffer. Only one UDP flow was required for the third scenario. Hosts `W1` and `E2` were used to generate the two UDP flows, both of which were destined to host `E1`. Different hosts were used to achieve high transfer rates. The ping flow sent messages from host `WA` to host `EA`. Therefore, contention for buffer and bandwidth resources occurrred on the link from router `WR` to router `ER`.

Our next step was to configure the routers. A single virtual queue was configured on the output interface from `WR` to `ER`, and all application flows were directed to this queue. In scenario 1, the whole link capacity was assigned to the virtual queue, but the buffer allocation was changed from 20% to 100%. In scenario 2, the assigned rate was varied from 1% to 100%, while the buffer allocation was set to 100%, and in scenario 3, the rate and buffer allocations were set to 20%, and the capacity sharing mode was explicitly configured.

Next, we executed the experiments corresponding to the scenarios. For the first two scenarios, each `nuttcp` application was initiated with the sending rate set to 7 Gbps, resulting in a total incoming rate of 14 Gbps in order to fill up the buffer of the 10 Gbps `WR`-to-`ER` interface. Due to the resulting packet losses, `nuttcp` at the receiving host `E1` reported rates of approximately 5 Gbps for each UDP flow. In scenario 3, the sending rate of the single UDP flow was set to 3 Gbps. This was sufficient given the 20% rate allocation to the configured virtual queue on the `WR`-to-`ER` link in this scenario. In all three scenarios, the UDP flows and ping flow were run for 60 seconds.

Finally, for the first and third scenarios, round-trip time (delay) measurements were obtained from the ping application on the `WA` host. For the second scenario, router counters for outgoing packets on the `WR`-to-`ER` link were read in order to find the number of packets transmitted within 60 seconds under different rate allocations.

2) Results and discussion:

Router buffer size: The ping packet delay measured in scenario 1 is plotted against the ping packet number, which is effectively the same as time, in Figure 3. With increasing time, the ping delay increases gradually because the `nuttcp` UDP packets start filling the buffer partition allocated to the virtual queue on the `WR`-to-`ER` interface. The minimum ping delay (2.1 ms) was observed when there were no UDP flows, i.e., there was no background traffic. The maximum delay (102 ms) was observed when the buffer allocation for the virtual queue was 100%.

In the various plots of Figure 3, the buffer allocations for the virtual queue are indicated. When the buffer allocation was limited to 20%, the delay was only 22.2 ms, while when the buffer allocation was set to 100%, the ping delay was higher because the whole buffer had filled up. Recall that the aggregate arrival rate of packets destined for the `WR`-to-`ER` link was 14 Gbps, while the outgoing link rate was only 10 Gbps.

Based on these observations, the buffer size for the `WR`-to-`ER` egress interface can be computed as follows:

$$10\ Gbps \times (102 - 2.1)\ ms = 125\ MB \qquad (1)$$

Default mode for link capacity sharing: From the experiments conducted in Scenario 2, the router counters for the `WR`-to-`ER` were recorded, and are shown in Table I. The reported packets were almost the same for all values of the link capacity allocation. Recall that the buffer allocation was set to 100% for this scenario. In other words, even if only a 1% rate was assigned to the virtual queue in which packets from all three flows were held, the virtual queue was served at 100% capacity. This result verifies that the default mode of operation for the tested router is the work-conserving mode.

Comparison of the two rate sharing modes: Two rate sharing strategies, strictly partitioned and work conserving,

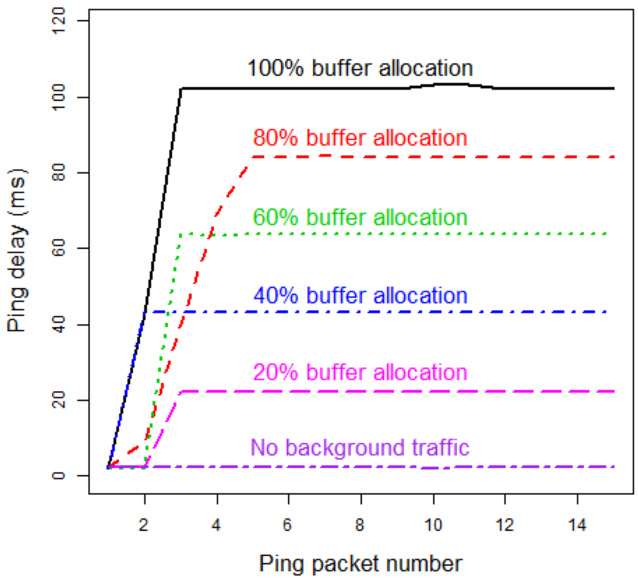


Figure 3. Experiment 1 scenario 1 results: Ping delay for different buffer allocations (rate allocation was 100%).

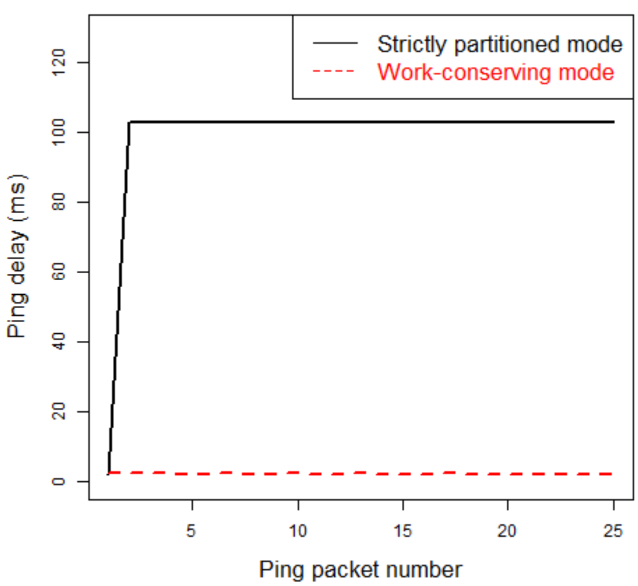


Figure 4. Experiment 1 scenario 3: Results comparing the two rate sharing modes (rate and buffer allocation was 20%).

were compared in Scenario 3. Figure 4 shows the ping delay results under these two configurations. In the strictly partitioned configuration, ping delays built up to 102 ms. Recall that for scenario 3, the virtual queue rate and buffer allocations were set to 20%, which was confirmed as follows:

$$R = \frac{125\ MB \times 0.2}{(102 - 2.1)\ ms} = 2\ Gbps \tag{2}$$

Under the work-conserving configuration, ping delay was only 2.1 ms (the round-trip time with no background traffic). Recall that the UDP flow sending rate was 3 Gbps in this scenario, while the rate allocation was only 2 Gbps. Yet there was no queue buildup in the buffer, which means the egress interface was served at a rate greater than 3 Gbps. Thus, in the work-conserving mode, virtual queues that have packets are served with excess capacity, if any.

C. Experiment 2

1) Purpose and execution: The goals of this experiment were to (i) determine whether α flows have adverse effects on real-time flows, and (ii) determine whether a scheduling-only no-policing solution of α-flow isolation to a separate virtual queue is sufficient to meet the first goal of keeping non-α flow delay/jitter low.

The first step was to plan a set of applications. We decided to use four `nuttcp` TCP flows and a ping flow. The TCP version used was H-TCP [20] because it is the recommended option to create high-speed (α) flows [21]. Two of the TCP flows carried data from host `E2` toward host `W1`, while the other two TCP flows were from `E1` to `W1`. The ping flow was from `EA` to `W1`. Therefore, in this experiment, contention for buffer and bandwidth resources occurred on the link from router `WR` to host `W1`. Although the high-performance host `W1` was the common receiver for all flows, there was no contention for CPU resources at `W1` because the operating system automatically scheduled the five receiving processes to different cores.

The second step was to configure the routers. For comparison purposes, this experiment required two configurations: (i) `1-queue`: a single virtual queue was defined on the egress interface from `WR` to `W1`, and all flows were directed to this queue, and (ii) `2-queues`: two virtual queues (α queue and β queue) were configured on the egress interface from `WR` to `W1`, and WFQ scheduling was enabled with the following rate (and buffer) allocations: 95% for α queue and 5% for β queue. The priority levels of the α and β virtual queues were set to medium-high and medium-low, respectively. In the 2-queues configuration, two additional steps were required. A firewall filter was created in router `WR` to identify packets from TCP (α) flows using their source and destination IP addresses. A

TABLE I. EXPERIMENT 1 SCENARIO 2: PACKET COUNTER VALUES OBSERVED AT ROUTER `WR` FOR ITS `WR`-TO-`ER` INTERFACE.

Link rate allocation	1%	20%	40%	60%	80%	100%
Number of packets transmitted on the `WR`-to-`ER` link	51924370	51536097	52755553	52669911	52786301	52637553

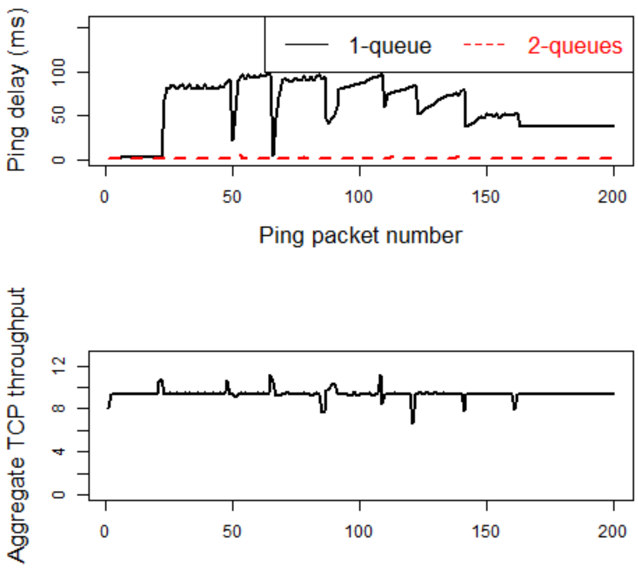


Figure 5. Experiment 2: Top graph shows the delays experienced in the ping flow under 1-queue and 2-queues configurations; bottom graph shows the aggregate TCP flow throughput.

class-of-service configuration command was used to classify these packets as belonging to the α class and to direct packets from these flows to the α queue on the egress interface from WR to W1. By default, all other packets were directed to the β queue, which means that packets from the ping flow were sent to the β queue.

In the third step, the applications were executed as follows. The four TCP flow execution intervals were: (0, 200), (20, 160), (40, 140), and (60, 120), respectively, while the ping flow was executed from 0 to 200 seconds.

Finally, throughput measurements as reported by each nuttcp sender were collected, as were the delays reported by the ping application.

2) Results and discussion: The top graph in Figure 5 illustrates that the scheduling-only no-policing solution of configuring two virtual queues on the shared egress interface and separating out the α flows into their own virtual queue leads to reduced packet delay/jitter for the β flow. In the 1-queue configuration, the mean ping delay was 60.4 ms, and the standard deviation was 29.3 ms, while in the 2-queues configuration, the mean ping delay was only 2.3 ms, with a standard deviation of 0.3 ms.

In the 2-queues case, since the rate of the ping flow was much lower than the 5% allocated rate for the β queue, the β queue was in-profile, and hence the ping-application packets were served immediately without incurring any queueing delays.

The bottom graph in Figure 5 shows the aggregate throughput of the four TCP flows. A comparison of this throughput graph with the top ping-delay graph shows the following: (i) when the aggregate TCP throughput increased from 9.4 Gbps to 10.7 Gbps at time 22, and the ping delay increased from 3 ms to 82 ms. The nuttcp application reports average throughput on a per-sec basis. Therefore, while the total instantaneous throughput cannot exceed 10 Gbps (link rate), the sum of the per-sec average throughput values for the four TCP flows sometimes exceeds 10 Gbps, (ii) when the aggregate TCP throughput dropped from 10.6 Gbps to 9.3 Gbps at time 49, the ping delay dropped from 92 ms to 22 ms, correspondingly, and (ii) throughput drops at 85, 121, 141, and 161 sec coincided with ping-delay drops.

D. Experiment 3

1) Purpose and execution: The goals of this experiment were to (i) compare a 2-queues configuration (scheduling-only, no-policing) with a 3-queues configuration (scheduling and policing), and (ii) compare multiple 3-queues configurations with different parameter settings.

As per our execution methodology, the first step was to plan applications. To study the behavior of the QoS mechanisms, one nuttcp TCP flow and one nuttcp UDP flow (background traffic) were planned. The UDP flow carried data from host E2 toward host W1, while the TCP flow was from E1 to W1. Contention for buffer and bandwidth resources occurred on the link from router WR to host W1.

In the second step, the router WR was configured with the following QoS mechanisms. The 2-queues configuration was the same as in Experiment 2 (no-policing), except that both queues were given equal weight in sharing the rate and buffer (50% each). For the 3-queues configurations, the allocations for the three queues (α, β, and *SS*) to which in-profile TCP-flow packets, UDP and ping packets, and out-of-profile TCP-flow packets, were directed, respectively, are shown in Table II. The priority levels of these three virtual queues were medium-high, medium-low, and low respectively. The policer was configured to direct in-profile TCP-flow packets ($\leq$ 1 Gbps and burst-size $\leq$ 31 KB) to the α queue, and out-of-profile packets to the *SS* queue.

In the third step, experiment execution, the UDP flow rate was varied from 0 Gbps to 3 Gbps in a particular on-off pattern as shown in the top graph of Figure 6, and the TCP flow was executed for the whole 200 sec. Finally, the same performance metrics were collected as in Experiment 2.

2) Results and discussion: Figure 6 shows the TCP throughput under the four configurations (one 2-queues and three 3-queues) for different rates of the background UDP flow. When the UDP flow rate was non-zero, since some of the plots overlap, we have summarized the mean TCP-flow throughput in Table II. When there was no background UDP traffic, the throughput of the TCP flow was around 9.1 Gbps for all four configurations as seen in the first row of Table II. As the background traffic load was increased, the throughput of the TCP flow in all the 3-queues configurations dropped more rapidly than in the 2-queues configuration, e.g., when the background UDP-flow rate was 3 Gbps, the TCP throughput was in the 300-610 Mbps range for the 3-queues

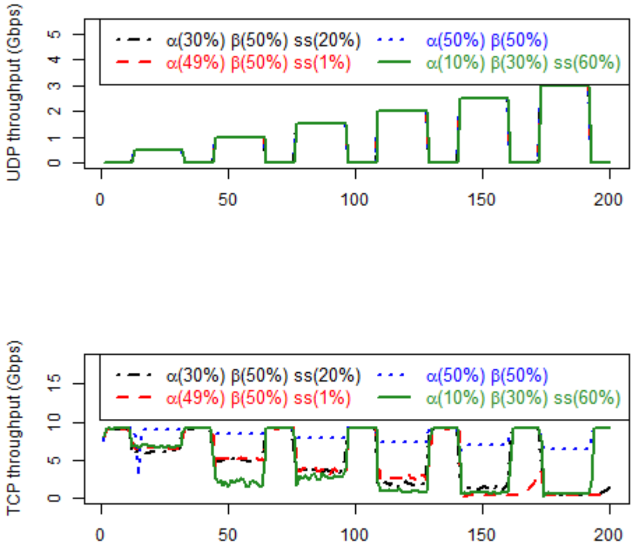


Figure 6. Experiment 3: The x-axis is time measured in seconds; the top graph shows the on-off mode in which the UDP rate was varied; the lower graph shows the TCP flow throughput under the four configurations.

TABLE II. EXPERIMENT 3: α-FLOW THROUGHPUT UNDER DIFFERENT BACKGROUND LOADS (UDP RATE) AND QOS CONFIGURATIONS.

UDP rate (Gbps)	α-flow throughput (Gbps)			
	Percentages for 2-queues (α, β) and 3-queues (α, β, *SS*) configurations			
	(50,50)	(49,50,1)	(30,50,20)	(10,30,60)
0	9.12	9.09	9.07	9.12
0.5	8.92	6.62	6.06	6.83
1	8.43	5.22	5	2.12
1.5	7.94	3.78	3.67	2.82
2	7.44	2.7	1.93	0.92
2.5	6.95	0.33	1.38	0.69
3	6.46	0.34	0.38	0.61

configurations, while the TCP throughput was 6.5 Gbps for the 2-queues scenario (see last row of Table II).

In addition to explaining the first and last rows of Table II, we provide an explanation for the drop in TCP-flow throughput in the last column of the row corresponding to UDP rate of 1 Gbps, which highlights the importance of choosing the WFQ allocations carefully.

Explanation for the first row of Table II: The explanation for the TCP-flow throughput when there was no background traffic is straightforward in the 2-queues configuration. As there were no packets to be served from the β queue and the transmitter was operating in a working-conserving manner, the β queue's 50% allocation was used instead to serve the α queue, and correspondingly the TCP flow enjoyed the full link capacity.

The explanation for the TCP-flow throughput values observed in the 3-queues configurations requires an understanding of the packet arrival pattern to the policer (see Figure 2) and the rate at which packets leave the policer. When TCP-flow throughput was almost the line rate (over 9 Gbps), then the rate at which in-profile packets left the policer was almost constant at 1 Gbps. This is because the token generation rate was 1 Gbps and packet inter-arrival times were too short for a significant collection of tokens in the bucket. Therefore, in an almost periodic manner, every tenth packet of the TCP flow was marked as being in-profile and sent to the α queue and the remaining 9 packets were classified as out-of-profile and sent to the *SS* queue. Given that in all the 3-queues configurations, the α queue was assigned at least 10% of the link rate/buffer space, the WFQ scheduler determined that the α queue was in-profile, and the PQ scheduler systematically served 1 packet from the α queue followed by 9 packets from the *SS* queue thus preserving the sequence of the TCP-flow packets. In the (49,50,1) configuration, 9 packets were served out of the SS queue in sequence even though the queue was out-of-profile after the first packet was served. This is because there were no packets in the β queue and none in the α queue given the policer's almost-periodic direction of 1-in-10 packets to this queue. Since no packets were out-of-sequence or lost, the TCP-flow throughput remained high at above 9 Gbps in all the 3-queues configurations.

Explanation for the last row of Table II: When there was background `nuttcp` UDP traffic at 3 Gbps, in the 2-queues configuration, it is easy to understand that the `nuttcp` TCP flow was able to use up most of the remaining bandwidth, which is the line rate minus the rate of background `nuttcp` UDP flow, and hence the TCP-flow throughput was about 6.5 Gbps.

The explanation for the low `nuttcp` TCP throughput in the 3-queues configurations is that the opposite of the systematic behavior explained above for the first row occurred here. When the incoming packet rate to the policer was lower than the line rate, the token bucket had an opportunity to collect a few tokens. Therefore, when TCP-flow packets arrived at the policer, a burst of them was classified as in-profile (since for every token present in the bucket, one packet is regarded as being in-profile), and sent to the α queue. These were served in sequence, but because the transmitter had to serve the β queue (for the UDP flow), the pattern in which the policer sent packets to the α queue and *SS* queue is unpredictable and involved bursts. This resulted in TCP segments arriving out-of-sequence at the receiver (as confirmed with `tcpdump` and `tcptrace` analyses presented in the next section). Out-of-sequence arrivals trigger TCP's Fast retransmit/Fast recovery algorithm, which causes the sender's congestion window to halve resulting in lower throughput.

Explanation for the last-column entry in the row corresponding to 1 Gbps in Table II: The TCP-flow throughput dropped much faster from 6+ Gbps to 2.12 Gbps when UDP rate increased from 0.5 to 1 Gbps in the (10,30,60) 3-queues

TABLE III. EXPERIMENT 4: QOS CONFIGURATIONS; OOP: OUT-OF-PROFILE.

Configuration	Policing	WFQ allocation `2-queues`:(α,β) `3-queues`:(α,β,`SS`)	WRED
`2-queues`	None	(60,40)	NA
`3-queues + policing1`	OOP to `SS` queue	(59,40,1)	NA
`3-queues + policing2`	OOP to `SS` queue	(20,40,40)	NA
`2-queues + policing + WRED`	WRED	(60,40)	Drop prob. = queue occ.

configuration than in the other two 3-queues configurations. This is explained using the above-stated reasoning that when the TCP-flow packets do not arrive at close to the line rate, the inter-packet arrival gaps allow the token bucket to collect a few tokens, making the policer send bursts of packets to the α queue. In this (10,30,60) configuration, after serving only one packet from each burst, the WFQ scheduler found the α queue to be out-of-profile since its allocation was only 10% or equivalently 1 Gbps. This led to a greater number of out-of-sequence arrivals at the TCP receiver than in the other two 3-queues configurations, and hence lower throughput.

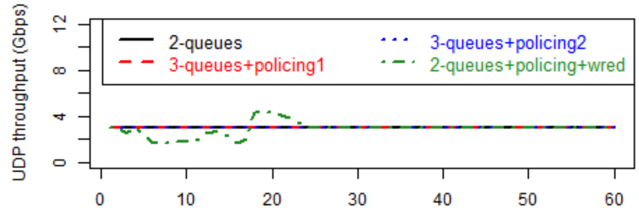
In *summary*, the higher the background traffic load, the lower the `nuttcp` TCP-flow packet arrival rate to the policer, the larger the inter-arrival gaps, the higher the number of collected tokens in the bucket, and the larger the number of in-profile packets directed to the α queue. If the WFQ allocation to the α queue is insufficient to serve in-profile bursts, packets from the α queue and *SS* queue will be intermingled resulting in out-of-sequence packets at the receiver. This fine point notwithstanding, the option of directing out-of-profile packets from the policer to a separate queue appears to be detrimental to α-flow throughput. We conclude that the second goal of high α-flow throughput cannot be met with this policing approach. In the next experiment, a different mechanism for dealing with out-of-profile packets was tested.

E. Experiment 4

1) Purpose and execution: The goal of this experiment was to compare the approach of applying WRED to out-of-profile packets rather than redirecting these packets to a scavenger-service queue as in Experiment 3. The planned applications were the same as in Experiment 3, i.e., to generate one `nuttcp` TCP flow and one `nuttcp` UDP flow.

The next step was router configuration. Four configurations are compared as shown in Table III. In the fourth option, Out-of-Profile (OOP) packets are dropped probabilistically at the same rate as the fraction of α-queue occupancy. In other words, if the α queue has 50% occupancy, then 50% of the OOP packets are dropped on average. The policing rate and burst size settings were the same as in Experiment 3.

Both the TCP and UDP flows were executed for 200 sec,



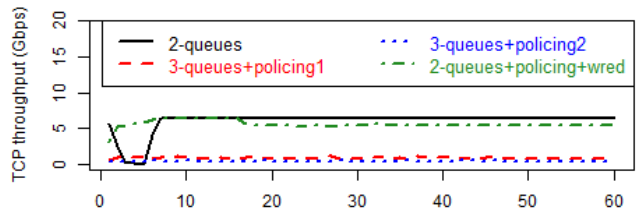


Figure 7. Experiment 4: The x-axis is time measured in seconds; the top graph shows the on-off mode in which the UDP rate was varied; the lower graph shows the TCP flow throughput under the four configurations.

but unlike in Experiment 3, the rate of the UDP flow was maintained unchanged at 3 Gbps for the whole time period. Finally, in addition to the previously used methods of obtaining throughput reports from `nuttcp`, two packet analysis tools, `tcpdump` and `tcptrace`, were used to determine the number of out-of-sequence packets at the receiver. Additionally, to find the number of lost packets, a counter was read at router `WR` for the `WR`-to-`W1` link before and after each application run.

2) Results and discussion: The lower graph in Figure 7 and Table IV show that the TCP-flow throughput is highest in the 2-queues (no-policing) scenario, with the WRED option close behind. The policing with WRED option performs much better than the options in which out-of-profile (OOP) packets are directed to an *SS* queue. In the WRED-enabled configuration, the TCP flow experiences a small rate of random packet loss, as shown in Table IV, while in 3-queues configurations, there were much higher numbers of out-of-sequence packets. The out-of-sequence packets in the WRED-enabled configuration result from the 15 lost packets, and are not independent events.

Surprisingly, even though the number of out-of-sequence packets was larger for the `3-queues+policing1` configuration, the throughput was higher in that configuration. This implies that fewer number of the out-of-sequence packets caused triple-duplicate ACKs in the first case. But this pattern is likely to change for repeated executions of the experiment.

Finally, Figure 7 shows that in the `2-queues` (no-policing) configuration, there was degradation of throughput soon after the flow started. Also, Table IV shows a loss of 5050 packets (the 4076 out-of-sequence packets were related to these

TABLE IV. EXPERIMENT 4: NUMBER OF OUT-OF-SEQUENCE PACKETS AND LOST PACKETS FOR DIFFERENT QOS SETTINGS.

Measure	2-queues	3-queues+ policing1	3-queues+ policing2	2-queues+ policing+wred
Average throughput	6 Gbps	0.92 Gbps	0.47 Gbps	5.6 Gbps
Num. of out-of-sequence packets at the receiver	4076	8812	7199	15
Num. of lost packets at the WR-to-W1 router link	5050	0	0	15

losses). Using tcptrace, we found that these losses occurred at the start of the transfer. This is explained by the aggressive growth of the congestion window (cwnd) in H-TCP, which uses a short throughput probing phase at the start. During the 1st second, the throughput of the TCP flow averaged 5.7 Gbps. The 5050 lost packets occurred in the 2^{nd} second. These losses occurred in the WR router buffer on its egress link from WR to W1. If H-TCP increased its cwnd to a large enough value to send packets at an instantaneous rate higher than 7 Gbps, then given the presence of the UDP flow at 3 Gbps, the α queue would fill up. From Experiment 1, we determined that the particular router used as WR has a 125 MB buffer. Since the buffer is shared between the α and β queues in a strictly partitioned mode with the 60-40 allocation, the α queue has 75 MB, which means that if the H-TCP sender exceeds the 7 Gbps rate by even 600 Mbps, the α queue will fill up within a second. Inspite of this initial packet loss, the 2-queues no-policing configuration achieves the highest throughput. In the next experiment, we consider the question of whether the use of policing and WRED has a fairness advantage when multiple α flows share a queue.

F. Experiment 5

1) Purpose and execution: The goal of this experiment was to understand how two α flows compete for bandwidth under different 2-queues configurations: without policing (2-queues), and with policing and WRED (2-queues+policing+WRED). In a first scenario, the α flows had similar round-trip times (RTTs), while in a second scenario, the RTTs differed significantly. We expected a fairness advantage for the policing/WRED scheme, but found the opposite. This is because neither of the two policed α flows honored their assigned rates, and while under the no-policing scheme the TCP flows adjusted their sending rates and had no packet losses, the deliberate packet losses in the policing/WRED scheme lowered throughput and resulted in a lower fairness. Thus, the no-policing configuration outperforms the configuration with policing and WRED from both throughput and fairness considerations when neither flow honors the policed rate.

The first step was to choose applications. Two nuttcp TCP flows were planned. The first TCP flow (TCP1) was from host E2 to host W1, and the second TCP flow (TCP2) was from host E1 to host W1. The RTTs were similar but not exactly the same. The RTT was 1.98 ms on the E2-to-W1 path and 2.23 ms on the E1-to-W1 path, because the latter path passes through an additional router, ER.

The router configurations were as follows. In the 2-queues configuration, packets from both TCP flows were directed to an α queue, with the rate and buffer allocations set to (60,40) for the α and β queues, respectively. In the 2-queues+policing+WRED configuration, the policing rate/burst size settings were the same as in Experiment 3, and Out-of-Profile (OOP) packets were dropped probabilistically with the same settings as in Experiment 4 ([0,100] drop probability corresponding to [0,100] buffer occupancy.

TCP1 and TCP2 execution intervals were (0, 200) and (51, 151), respectively. In the different-RTTs scenario, the RTT of TCP2 was increased by 50 ms using tc. Finally, throughput and retransmission data were collected every second by nuttcp at the senders.

2) Results and discussion: Experimental results are presented for the similar-RTT and different-RTTs scenarios.

Similar-RTT scenario:

Figure 8 shows the throughput of the two TCP flows when they compete for the bandwidth and buffer resources of the α queue. In the 2-queues configuration, the throughput of TCP1 was approximately 9.4 Gbps for the first 50 seconds, but dropped to 7.1 Gbps at $t = 51$, since TCP2 was initiated then. In the 52^{nd} second, both flows suffered packet losses, with TCP1 requiring 2418 retransmissions and TCP2 requiring 3818 retransmissions. Since the sum of the rates of the flows exceeded 10 Gbps, it caused losses and retransmissions in the 52^{nd} sec. After the 52^{nd} second, there were no retransmissions on either flow. The per-second throughput recorded by nuttcp, from $t = 51$ to $t = 151$ during which both TCP flows were active, is shown in Figure 9. As the buffer filled up and queueing delays increased, TCP acknowledgments (ACKs) would have been delayed causing RTT for TCP1 to increase. This decreased the effective sending rate (cwnd/RTT). No losses occurred in the rest of the experiment because as sending rates increased in one or both flows, the buffer filled up delaying packets, and hence increasing RTT, which in turn caused the sending rate to drop thus reducing buffer buildups. This oscillatory behavior can be observed in the throughput sum plot of Figure 9. The higher rate of TCP1 could be because of the slightly lower RTT for this flow when

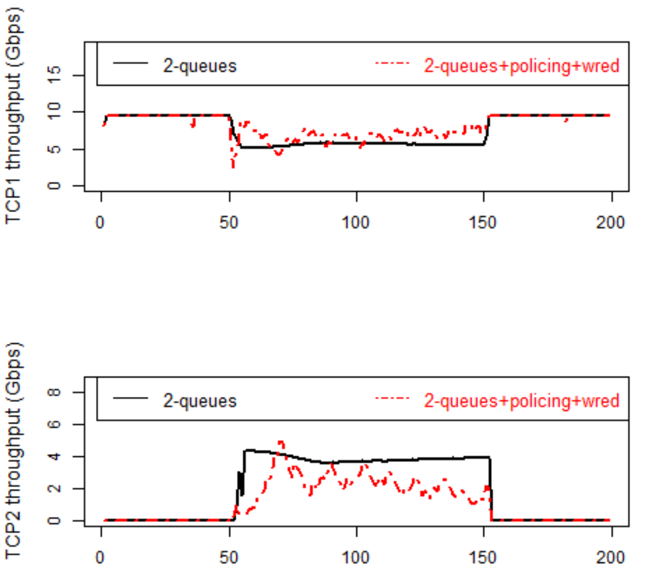


Figure 8. Experiment 5: Throughput of two TCP flows under two QoS configurations (similar RTTs).

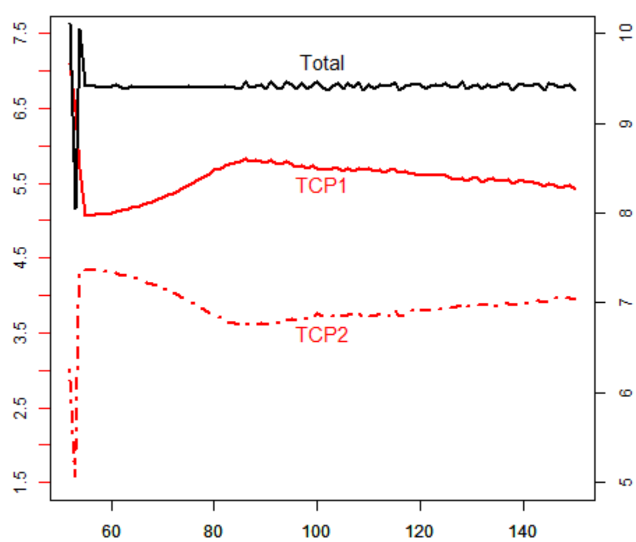


Figure 9. Experiment 5: Throughput of two TCP flows, and their total throughput in the `2-queues` configuration (similar RTTs).

compared to that of TCP2.

TABLE V. EXPERIMENT 5: RETRANSMISSIONS AND THROUGHPUT OF 2 TCP FLOWS FOR THE POLICING/WRED CONFIGURATION (SIMILAR RTTs).

Time (s)	TCP	Retrans-missions	Throughput (Gbps)		
			Min	Median	Max
51 - 53	TCP1	227	2.56	4.84	6.31
	TCP2	32	0.46	0.5	1.03
54 - 69	TCP1	14	4.35	7.37	8.98
	TCP2	0	0.47	1.78	4.98
70 - 151	TCP1	65	4.26	6.91	8.47
	TCP2	3	1.02	2.26	4.26

Next, consider the throughput values of `TCP1` and `TCP2` in the `2-queues+policing+WRED` configuration shown in Figure 8. From $t = 51$, `TCP1` suffered losses and its throughput dropped steadily until it reached 4.35 Gbps, while `TCP2` throughput kept increasing until it reached 4.98 Gbps at $t = 69$. The reason why `TCP1` throughput dropped is because of the policing limit of 1 Gbps. Packets exceeding this rate were marked as out-of-profile. Since `TCP1` rate was 9.4 Gbps at $t = 50$ just before `TCP2` was started, its sending rate was well above the policing rate of 1 Gbps. Subsequent to reaching this almost balanced throughput level at $t = 69$, losses, and hence retransmissions, were observed on both flows, but there were more losses in `TCP1` (see Table V) because its rate was higher.

The key difference between the `2-queues` and `2-queues+policing+WRED` configurations is that there were no losses in the former configuration after $t = 53$, while in the latter configuration both flows kept experiencing packet losses. This is because in the second configuration, as both flows exceeded the policing limit of 1 Gbps, a few packets were marked as out-of-profile in both flows. Recall that under WRED packets are dropped probabilistically at a rate equal to buffer occupancy, and since the buffer will sometimes have packets, losses are inevitable in the `2-queues+policing+WRED` configuration. When losses occurred under the `2-queues+policing+WRED` configuration, the slight edge in RTT for `TCP1` may account for its higher throughput when compared to `TCP2`. `TCP1` maintained an average rate of 6.86 Gbps from $t = 70$ to $t = 151$ when `TCP2` was terminated, at which point `TCP1` recovered its rate to 9.4 Gbps. The `TCP2` average throughput from $t = 70$ to $t = 151$ was smaller at 2.35 Gbps. A loss detected with triple duplicate ACKs results in a halving of `cwnd`, which is equivalent to halving the sending rate. `TCP1` operated in a higher range of `cwnd` values when compared to `TCP2`.

Using Jain's fairness index [22],

$$f(x) = \frac{(\sum_{i=1}^{n} x_i)^2}{n \cdot \sum_{i=1}^{n} x_i^2} \quad x_i \geq 0 \tag{3}$$

and average throughput values across the $t = 51$ to $t = 151$ time range, we computed the fairness values to be 0.97 and

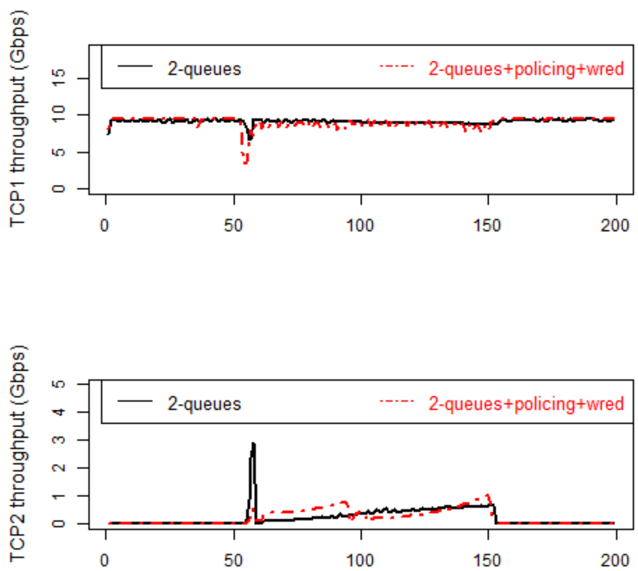


Figure 10. Experiment 5: Throughput of two TCP flows under two QoS configurations (different RTTs).

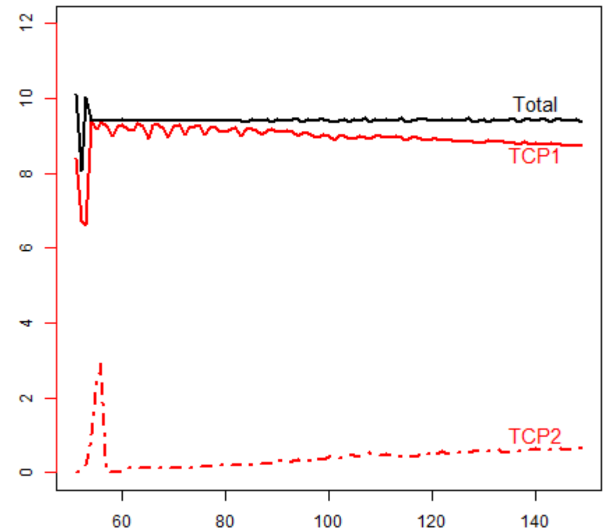


Figure 11. Experiment 5: Throughput of two TCP flows, and their total throughput in the `2-queues` configurations (different RTTs).

0.8 for the `2-queues` and `2-queues+policing+WRED` configurations, respectively. This does not imply that the former is a more fair configuration; it is just that in this experiment, given that both TCP flows did not honor the policing limit, policing caused packet losses, and recovery from packet losses was slower for the longer-RTT path even if the RTT difference was small. Without policing, there were no deliberate packet drops in the `2-queues` configuration; instead the TCP senders self-regulated their sending rates. When the rates were high, buffer occupancy grew, but this caused RTT to increase, which, in turn, caused a lower sending rate.

In *summary*, this experiment showed that policing will result in decreased throughput for TCP based α flows when two or more such flows occur simultaneously. In Experiment 4, policing with WRED did not impact throughput significantly but there was only one TCP based α flow, unlike in this experiment.

Different RTTs:

Figure 10 shows the throughput of the two TCP flows with different RTTs. During the 100-second period when both TCP flows were active, the throughput of the two TCP flows and their total throughput are plotted in Figure 11. The throughput of `TCP1` dropped from 9.1 Gbps at $t = 50$ to 7.1 Gbps at $t = 51$, since `TCP2` was initiated at time 50. In the 57^{th} second, when `TCP2` built up its rate to 2.93 Gbps, which made the sum of the rates exceed 10 Gbps, both flows suffered packet losses, with `TCP1` requiring 3315 retransmissions and `TCP2` requiring 4118 retransmissions. After the 57^{th} second, there were no retransmissions on either flow. Since the RTT of `TCP2` was increased by 50 ms, it took 6 sec to reach the time instant when losses occurred unlike in the similar-RTT scenario in which both flows experienced losses in 2 sec. In the second after the losses, `TCP1` recovered its throughput back to 9.38 Gbps, while `TCP2` throughput decreased from 2.92 Gbps to 11 Mbps.

TABLE VI. EXPERIMENT 5: RETRANSMISSIONS AND THROUGHPUT OF 2 TCP FLOWS FOR THE POLICING-WRED CONFIGURATION (DIFFERENT RTTs).

Time (s)	TCP	Retrans-missions	Throughput (Gbps)		
			Min	Median	Max
51 - 58	TCP1	140	3.51	7.6	9.41
	TCP2	1	0.003	0.086	0.54
59 - 151	TCP1	107	7.61	8.81	9.35
	TCP2	4	0.056	0.42	0.98

Next, we repeated the experiments with the policing and WRED configuration. The retransmissions and throughput of the two TCP flows are shown in Table VI. `TCP1` experienced losses even after the initial set of losses unlike in the `2-queues` configuration. Consequently, TCP1's average throughput was lower in the `2-queues+policing+WRED` configuration (8.98 Gbps) than in the `2-queues` configuration (9.1 Gbps), while TCP2's average throughput was higher (0.43 Gbps vs 0.41 Gbps). Jain's fairness index value for throughput of the two TCP flows was comparable under the two configurations (0.546 and 0.551 under the `2-queues` and `2-queues+policing+WRED` configurations, respectively). The RTT difference was the dominant reason for the unfair

TABLE VII. EXPERIMENT 6: UDP-FLOW LOSS RATE AND PING DELAY.

	β queue rate and buffer allocation	UDP rate (Gbps)	UDP flow average packet loss rate before, during, and after the TCP flow			Average ping delay (ms) before, during, and after the TCP flow		
			t ∈ (0-52)	t ∈ (53-153)	t ∈ (154-200)	t ∈ (0-52)	t ∈ (53-153)	t ∈ (154-200)
Reno	20%	2	0	5.03%	0	2.25	103	2.26
	30%	2	0	0	0	2.3	2.25	2.25
	20%	3	0	39.33%	0	2.26	103	2.27
	30%	3	0	4.57%	0	2.27	104	2.31
	$\geq$ 30%	2 or 3	0	0	0	2.26	2.26	2.26
H-TCP	20%	2	0	5.3%	0	2.27	103	2.27
	30%	2	0	0	0	2.27	2.26	2.25
	20%	3	0	39.3%	0	2.26	103	2.27
	30%	3	0	4.67%	0	2.28	104	2.29
	$\geq$ 30%	2 or 3	0	0	0	2.26	2.27	2.27

treatment of `TCP2`, not the QoS configuration.

G. Experiment 6

1) Purpose and execution: The goals of this experiment were to (i) identify the impact of QoS provisioning under changing traffic conditions, and (ii) compare two versions of TCP: Reno and H-TCP. In the first part, we studied the effect of enabling QoS control, specifically, the `2-queues` configuration, on changing traffic patterns. For example, what is the impact of background traffic increasing to 3 Gbps when the β queue to which background traffic was directed was allocated only 20% of the rate/buffer capacity on a 10 Gbps link (based on previous traffic measurements). As α flows occur infrequently, most of the time, service quality for the background traffic would be unaffected, but if this surge in background traffic occurred within the duration of an α flow, there could potentially be higher losses and delays in the background traffic than if QoS mechanisms had not been enabled.

As mentioned in Section III-C, the TCP version used in our experiments was H-TCP, the recommended option for high-speed networks [21]. However, although computers dedicated for high-speed transfers are likely to be configured to use H-TCP, as the most widely used TCP version is still TCP Reno, we undertook a comparative experiment.

Three applications were planned for this experiment: one `nuttcp` TCP flow (from host `E1` to `W1`), one `nuttcp` UDP flow (from host `E2` to `W1`) and one ping flow (from host `EA` to `W1`). In the router configuration step, two queues were configured: a β queue for the background UDP flow and the ping flow, and an α queue for the TCP flow. The rate/buffer allocation (the same percentage was used for both resources) for the β queue was varied from 20% to 60% in 10% increments, and the allocations for the α queue were set correspondingly. The applications were executed as follows: UDP flow and ping flow in the time interval (0, 200), and the TCP flow in the interval (53, 153). Two rates were used for the UDP flow: 2 Gbps and 3 Gbps.

2) Results and discussion:

Goal 1: Table VII shows the UDP-flow loss rate and ping delay under different rate/buffer allocations for the β queue in the `2-queues` configuration. Before the TCP flow was initiated (the first 53 seconds) and after the TCP flow ends (the last 47 seconds), even if the rate of the UDP flow exceeded the allocated rate for the β queue (i.e., 20% allocation when the UDP-flow rate was 3 Gbps), the UDP flow experienced no losses, and the ping delay remained at around 2.26 ms, which implies that there was no buffer buildup in the β queue. This is because the transmitter was operating in work-conserving mode, which allowed it to serve packets from the β queue as the α queue was empty.

During the time interval (53-153) when the TCP flow was active, with a 20% rate/buffer allocation for the β queue, a 2 Gbps UDP flow suffered a 5% packet-loss rate, and the ping delay was 103 ms, which means the β queue was full. When the UDP-flow rate was increased to 3 Gbps, while the β-queue allocation was held at 20% (to model changing traffic conditions), the UDP-flow packet loss rate increased to about 39%, and the ping delay remained at 103 ms. Such a significant loss rate and increased packet delay would not have occurred had separate QoS classes not been created and the buffer not been divided. When the UDP-flow rate increased, the TCP-flow rate would have decreased as it would also have suffered losses. In the `2-queues` configuration, the TCP flow suffered no losses for both the combinations described above: 20% β queue allocation with 2 Gbps UDP-flow rate, and the 20%-3 Gbps combination. This is because the TCP flow was directed to the α queue, which had its own large (80%) buffer/rate allocation.

Goal 2: The numbers in Table VII show that there was no difference between H-TCP and Reno with regards to the impact of the TCP flow on the UDP and ping flows. Furthermore, Table VIII shows that the TCP flow enjoyed the same rate for most of its duration. When the background UDP-flow rate was 2 Gbps, the TCP-flow throughput was 7.45 Gbps, and when the UDP-flow rate was 3 Gbps, the TCP-flow throughput was correspondingly lower at 6.45 Gbps, irrespective of β-queue rate/buffer allocation. The only difference observed between Reno and H-TCP was in the TCP-flow's behavior in the first few seconds as shown in Table IX. Recall the TCP flow was started at $t = 53$. With Reno, the number of retransmissions that occurred in the early seconds drops as the β-queue buffer allocation was increased (and the α-queue size, to which the TCP flow was directed, correspondingly decreased). With smaller α-queue sizes, it appears that the TCP sender starts reducing its sending rate sooner, and hence there were fewer losses and retransmissions. We expected H-TCP to suffer more losses in the initial few seconds as it is more aggressive in increasing its sending window, but this was not observed. Both adjusted their sending rates and experienced no losses after the initial set of losses shown in Table IX.

TABLE VIII. EXPERIMENT 6: TCP-FLOW THROUGHPUT FOR MOST OF THE DURATION.

Background (UDP) rate	TCP throughput	
	Reno	H-TCP
2 Gbps	7.45 Gbps	6.45 Gbps
3 Gbps	7.45 Gbps	6.45 Gbps

TABLE IX. EXPERIMENT 6: TCP-FLOW RETRANSMISSIONS IN ITS FIRST FEW SECONDS (THE FLOW WAS STARTED AT $t = 53$).

UDP-flow rate	β-queue rate/ buffer alloc.	Time	Number of retx pkts
Reno			
2 Gbps	30%	$t = 54$	6624
	40%	$t = 54$	5811
	50%	$t = 53$	4327
	60%	$t = 54$	2645
3 Gbps	30%	$t = 54$	7673
	40%	$t = 54$	6970
	50%	$t = 53$	5137
	60%	$t = 54$	3495
H-TCP			
2 Gbps	30%	$t = 54$	6008
	40%	$t = 54\&55$	4322 & 298
	50%	$t = 53$	3825
	60%	$t = 54$	3966
3 Gbps	30%	NA	0
	40%	$t = 54$	1423
	50%	NA	0
	60%	$t = 54$	3528

H. Experiment 7

1) Purpose and execution: As described in Section II, HNTES uses an offline approach by analyzing NetFlow reports of completed flows to determine source-destination addresses of α flows, and then uses these addresses to configure firewall

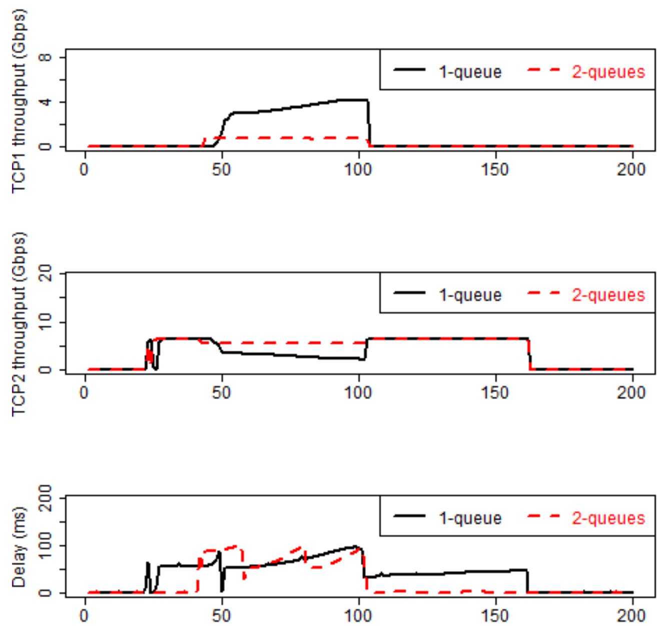


Figure 12. Experiment 7: The impact of an unidentified α flow with and without HNTES.

filters in ingress routers of a provider's network to redirect packets of future α flows to traffic-engineered QoS-controlled paths. With this scheme, an α flow between a new source-destination pair will not be identified as such until its NetFlow reports are analyzed, which most likely will occur after the flow completes. Such unidentified α flows will be directed to the β queue in a `2-queues` configuration. Since in such a configuration, buffer resources are partitioned between the β queue and α queue, the purpose of this experiment was to study the impact of such unidentified α flows.

Three `nuttcp` flows were planned for this experiment: a UDP flow from `E2` to `W1`, TCP flow `TCP1` from `E2` to `W1`, and a second TCP flow `TCP2` from `E1` to `W1`. In addition, a ping flow was executed from from `EA` to `W1`. Two router configurations were used in this experiment: (i) `1-queue`: a single virtual queue was defined on the egress interface from `WR` to `W1`, and all flows were directed to this queue, and (ii) `2-queues`: two virtual queues (α queue and β queue) were configured on the egress interface from `WR` to `W1`, and WFQ scheduling was enabled with the following rate (and buffer) allocations: 60% for α queue and 40% for β queue.

The execution intervals of the flows, ping, UDP, `TCP1`, and `TCP2` were (0,200), (0, 200), (42, 101), and (22, 162), respectively. The rate of the UDP flow was set to 3 Gbps. We assumed `TCP1` to be the unidentified α flow, which was hence directed to the β queue, while TCP2 was assumed to be an α flow from a previously seen source-destination pair, and hence directed to the α queue. The ping and UDP flows were directed to the β queue. Measurements were collected from the `nuttcp` and ping applications.

2) Results and discussion: The throughput of the two TCP flows and the ping delays are shown in Figure 12. In the `1-queue` configuration, during the 60 seconds when both TCP flows were active, `TCP1` throughput kept increasing to 4.16 Gbps, while `TCP2` throughput kept decreasing from 6.5 Gbps to 2.26 Gbps. This is because the RTT was slightly lower for `TCP1` as discussed earlier.

In the `2-queues` configuration, `TCP1` throughput was only 1 Gbps. This is because the β queue allocation was 40% of the link rate/buffer, of which 3 Gbps was used by the UDP flow, and `TCP2` was actively consuming the 60% allocation of the α queue. The mean throughput of the new α flow (TCP1) in the `1-queue` case was 3.2 Gbps, while it was only 0.8 Gbps under the `2-queues` configuration. In other words, the presence of HNTES and the corresponding `2-queues` configuration had an adverse effect on the unidentified α flow, though as shown in our prior work, most α-flow generating source-destination pairs send repeated α flows [3].

TABLE X. EXPERIMENT 7: TCP-FLOW RETRANSMISSIONS AND PING DELAYS.

(sec, no. of `TCP1` retx)	(sec, no. of) `TCP2` retx)	(sec, ping delay (ms))
`1-queue configuration`		
NA	(23, 3267)	(24, 2.25)
(50, 955)	(50, 568)	(50, 4.7)
`2-queues configuration`		
NA	(22, 8074)	(22, 2.3)
(44, 1855)	(44, 0)	(44, 87.3)
(60, 7)	(60, 0)	(60, 30.2)
(82, 7)	(82, 0)	(83, 48.5)

Consider the impact of the unidentified α flow on the ping flow. In the `1-queue` configuration, the ping delay was around 2.3 ms until `TCP2` was initiated at $t = 22$, at which instant the ping delay surged up to 65.9 ms as seen in Figure 12 because of the buffer build-up from TCP2 packets. Since H-TCP is aggressive in increasing its sending rate, in its 2^{nd} second ($t = 23$), there were 3267 packet drops as shown in Table X. With all these losses, ping delay correspondingly dropped down to 2.25 ms at $t = 24$. However, the delay quickly increased back to the 56 ms range peaking at 88.7 ms at $t = 49$. As shown in Table X, it took a few seconds after `TCP1` was initiated for both `TCP1` and `TCP2` to experience packet losses causing ping delay to drop back down to 4.7 ms at $t = 50$. Beyond this time instant, neither TCP flow suffered losses with both adjusting their sending rates based on received acknowledgments and ping delay peaked at 91.5 ms at $t = 101$ when `TCP1` ended. The ping delay dropped to 34 ms and increased to 47.9 ms at which point it dropped to 2.3 ms at $t = 162$ when `TCP2` ended.

In the `2-queues` configuration, the ping delay stayed around 2.3 ms even after `TCP2` was initiated as seen in Figure 12 (because `TCP2` was directed to a different queue), but increased to 87.3 ms when `TCP1` was initiated at $t = 43$ (since `TCP1` representing an unidentified α flow was directed to the same queue as the ping flow). `TCP2` suffered no losses after the initial losses of 8074 packets in its first second. On the other hand, `TCP1` suffered losses not only in its first second (1855 losses), but again at $t = 60$ and $t = 82$. During these seconds, ping delay dropped correspondingly from 103 ms at $t = 59$ to 30.2 ms at $t = 60$, and from 103 ms at $t = 82$ to 48.5 ms at $t = 83$. These results illustrate that the smaller buffer allocation for the β queue can have a negative effect on real-time flows when an unidentified α flow appears.

In summary, QoS partitioning does have negative effects when mismatched with traffic as shown in Experiment 6, and when α flows are undetected and hence handled by the partition set aside for β flows. Nevertheless, the benefits of QoS partitioning as illustrated in the first five experiments outweigh these costs.

IV. CONCLUSIONS AND FUTURE WORK

To reduce the impact of large-sized, high-rate (α) transfers on real-time flows, a Hybrid Network Traffic Engineering System (HNTES) was proposed in earlier work. HNTES is an intra-domain solution that enables the automatic identification of α flows at a provider network's ingress routers, and redirects these flows to traffic-engineered QoS-controlled virtual circuits. The purpose of this work was to determine the best QoS mechanisms for the virtual circuits used in this application. Our findings are that a no-policing, two-queues (one for α flows and one for β flows) solution with weighted fair queueing and priority queueing is both sufficient and the best for this application. It allows for the dual goals of reduced delay/jitter in real-time flows, and high-throughput for the α flows, to be met.

We studied two types of policing schemes for handling out-of-profile packets: redirection to a (third) scavenger-service (SS) queue and Weighted Random Early Detection (WRED) in which out-of-profile packets are either dropped probabilistically according to some profile or held in the same queue as in-profile packets. The WRED scheme was better than the SS-queue scheme because the latter caused out-of-sequence arrivals at the receiver, which triggered TCP congestion control mechanisms that led to lower throughput. However, the no-policing solution was better than the policing/WRED solution because in this application flows are not likely to honor the circuit rates and therefore deliberate packet drops are inevitable in the policing/WRED solution causing lowered throughput. The negatives of partitioning rate/buffer space resources between two queues were studied. Our conclusions are that close network monitoring is required to dynamically adjust the rate/buffer space split between the two queues as traffic changes, and the probability of unidentified α flows should be reduced whenever possible to avoid these flows from becoming directed to the β queue.

As future work, we plan to develop theoretical and/or simulation models to characterize the impact of QoS provisioning schemes on TCP throughput.

V. Acknowledgment

The University of Virginia portion of this work was supported by the U.S. Department of Energy (DOE) grant DE-SC0002350, DE-SC0007341 and NSF grants OCI-1038058, OCI-1127340, and CNS-1116081. The ESnet portion of this work was supported by the Director, Office of Science, Office of Basic Energy Sciences, of the U.S. DOE under Contract No. DE-AC02- 05CH11231. This research used resources of the ESnet ANI Testbed, which is supported by the Office of Science of the U.S. DOE under contract DE-AC02-05CH11231, funded through the American Recovery and Reinvestment Act of 2009.

References

[1] Z. Yan, M. Veeraraghavan, C. Tracy, and C. Guok, "On how to provision Quality of Service (QoS) for large dataset transfers," in *Proceedings of the Sixth International Conference on Communication Theory, Reliability, and Quality of Service (CTRQ)*, Apr. 21-26, 2013.

[2] S. Sarvotham, R. Riedi, and R. Baraniuk, "Connection-level analysis and modeling of nework traffic," in *ACM SIGCOMM Internet Measurement Workshop 2001*, Nov. 2001, pp. 99–104.

[3] Z. Yan, C. Tracy, and M. Veeraraghavan, "A hybrid network traffic engineering system," in *Proc. of IEEE 13th High Performance Switching and Routing (HPSR) 2012*, Jun. 24-27 2012.

[4] Esnet. Retrieved: 09.10.2013. [Online]. Available: http://www.es.net/

[5] GEANT. Retrieved: 09.10.2013. [Online]. Available: http://www.geant.net/

[6] JGN-X. Retrieved: 09.10.2013. [Online]. Available: http://www.jgn.nict.go.jp/english/

[7] On-Demand Secure Circuits and Advance Reservation System (OSCARS). Retrieved: 09.10.2013. [Online]. Available: http://www.es.net/services/virtual-circuits-oscars

[8] J. Spragins, "Asynchronous Transfer Mode: Solution for Broadband ISDN, Third Edition [New Books]," Jan./Feb. 1996, pp. 7.

[9] E. R. Braden, L. Zhang, S. Berson, S. Herzog, and S. Jamin, "Resource ReSerVation Protocol (RSVP)," RFC 2205, Sep. 1997.

[10] Z. Liu, M. Veeraraghavan, Z. Yan, C. Tracy, J. Tie, I. Foster, J. Dennis, J. Hick, Y. Li, and W. Yang, "On using virtual circuits for GridFTP transfers," in *The International Conference for High Performance Computing, Networking, Storage and Analysis 2012 (SC 2012)*, Nov. 10-16, 2012, pp. 81:1–81:11.

[11] GridFTP. Retrieved: 09.10.2013. [Online]. Available: http://globus.org/toolkit/docs/3.2/gridftp/

[12] A. Sniady and J. Soler, "Performance of Flow-Aware Networking in LTE backbone," in *Proceedings of OPNETWORK2012, OPNET 2012*.

[13] J. Kurose and K. Ross, "Computer networks: A top down approach featuring the internet," Pearson Addison Wesley, 2010.

[14] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance," *IEEE/ACM Transactions on Networking*, vol. 1, no. 4, pp. 397–413, Aug. 1993.

[15] D. Lin and R. Morris, "Dynamics of random early detection," in *ACM SIGCOMM Computer Communication Review*, vol. 27, no. 4, Oct. 1997, pp. 127–137.

[16] WRED. Retrieved: 09.10.2013. [Online]. Available: http://www.cisco.com/en/US/docs/ios/11_2/feature/guide/wred_gs.html

[17] R. Guérin, S. Kamat, V. Peris, and R. Rajan, "Scalable QoS provision through buffer management," in *ACM SIGCOMM Computer Communication Review*, vol. 28, no. 4, Oct. 1998, pp. 29-40.

[18] R. Pan, L. Breslau, B. Prabhakar, and S. Shenker, "Approximate fairness through differential dropping," *ACM SIGCOMM Computer Communication Review*, vol. 33, no. 2, pp. 23–39, Apr. 2003.

[19] R. P. Vietzke. (2008, Aug.) Internet2 headroom practice. Retrieved: 09.10.2013. [Online]. Available: https://wiki.internet2.edu/confluence/download/attachments/17383/Internet2+Headroom+Practice+8-14-08.pdf

[20] D. Leith and R. Shorten, "H-TCP: TCP for high-speed and long-distance networks," in *Protocols for Fast Long Distance Networks Workshop (PFLDnet)*, Feb. 16-17, 2004.

[21] Retrieved: 09.10.2013. [Online]. Available: http://fasterdata.es.net/host-tuning/linux/

[22] R. Jain, D. Chiu, and W. Hawe, "A quantitative measure of fairness and discrimination for resource allocation in shared computer systems," *Technical Report TR-301, DEC Research*, Sep. 1984.

www.iariajournals.org

International Journal On Advances in Intelligent Systems
ICAS, ACHI, ICCGI, UBICOMM, ADVCOMP, CENTRIC, GEOProcessing, SEMAPRO, BIOSYSCOM, BIOINFO, BIOTECHNO, FUTURE COMPUTING, SERVICE COMPUTATION, COGNITIVE, ADAPTIVE, CONTENT, PATTERNS, CLOUD COMPUTING, COMPUTATION TOOLS, ENERGY, COLLA, IMMM, INTELLI, SMART, DATA ANALYTICS
issn: 1942-2679

International Journal On Advances in Internet Technology
ICDS, ICIW, CTRQ, UBICOMM, ICSNC, AFIN, INTERNET, AP2PS, EMERGING, MOBILITY, WEB
issn: 1942-2652

International Journal On Advances in Life Sciences
eTELEMED, eKNOW, eL&mL, BIODIV, BIOENVIRONMENT, BIOGREEN, BIOSYSCOM, BIOINFO, BIOTECHNO, SOTICS, GLOBAL HEALTH
issn: 1942-2660

International Journal On Advances in Networks and Services
ICN, ICNS, ICIW, ICWMC, SENSORCOMM, MESH, CENTRIC, MMEDIA, SERVICE COMPUTATION, VEHICULAR, INNOV
issn: 1942-2644

International Journal On Advances in Security
ICQNM, SECURWARE, MESH, DEPEND, INTERNET, CYBERLAWS
issn: 1942-2636

International Journal On Advances in Software
ICSEA, ICCGI, ADVCOMP, GEOProcessing, DBKDA, INTENSIVE, VALID, SIMUL, FUTURE COMPUTING, SERVICE COMPUTATION, COGNITIVE, ADAPTIVE, CONTENT, PATTERNS, CLOUD COMPUTING, COMPUTATION TOOLS, IMMM, MOBILITY, VEHICULAR, DATA ANALYTICS
issn: 1942-2628

International Journal On Advances in Systems and Measurements
ICQNM, ICONS, ICIMP, SENSORCOMM, CENICS, VALID, SIMUL, INFOCOMP
issn: 1942-261x

International Journal On Advances in Telecommunications
AICT, ICDT, ICWMC, ICSNC, CTRQ, SPACOMM, MMEDIA, COCORA, PESARO, INNOV
issn: 1942-2601