International Journal on

Advances in Intelligent Systems

2010 vol. 3 nr. 1&2

The *International Journal on Advances in Intelligent Systems* is Published by IARIA.
ISSN: 1942-2679
journals site: http://www.iariajournals.org
contact: petre@iaria.org

Sponsored by IARIA
www.iaria.org

International Journal on Advances in Intelligent Systems
Volume 2, Number 4, 2009

- Thomas C. Schmidt, University of Applied Sciences – Hamburg, Germany
- Karolj Skala, Rudjer Bokovic Institute - Zagreb, Croatia
- Chieh-yih Wan, Intel Corporation, USA
- Hoo Chong Wei, Motorola Inc, Malaysia

Ubiquitous Systems and Technologies

- Matthias Bohmer, Munster University of Applied Sciences, Germany
- Dominic Greenwood, Whitestein Technologies AG, Switzerland
- Arthur Herzog, Technische Universitat Darmstadt, Germany
- Reinhard Klemm, Avaya Labs Research-Basking Ridge, USA
- Vladimir Stantchev, Berlin Institute of Technology, Germany
- Said Tazi, LAAS-CNRS, Universite Toulouse 1, France

Advanced Computing

- Dumitru Dan Burdescu, University of Craiova, Romania
- Simon G. Fabri, University of Malta – Msida, Malta
- Matthieu Geist, Supelec / ArcelorMittal, France
- Jameleddine Hassine, Cisco Systems, Inc., Canada
- Sascha Opletal, Universitat Stuttgart, Germany
- Flavio Oquendo, European University of Brittany - UBS/VALORIA, France
- Meikel Poess, Oracle, USA
- Said Tazi, LAAS-CNRS, Universite de Toulouse / Universite Toulouse1, France
- Antonios Tsourdos, Cranfield University/Defence Academy of the United Kingdom, UK

Centric Systems and Technologies

- Razvan Andonie, Central Washington University - Ellensburg, USA / Transylvania University of Brasov, Romania
- Kong Cheng, Telcordia Research, USA
- Vitaly Klyuev, University of Aizu, Japan
- Josef Noll, ConnectedLife@UNIK / UiO- Kjeller, Norway
- Willy Picard, The Poznan University of Economics, Poland
- Roman Y. Shtykh, Waseda University, Japan
- Weilian Su, Naval Postgraduate School - Monterey, USA

GeoInformation and Web Services

- Christophe Claramunt, Naval Academy Research Institute, France
- Wu Chou, Avaya Labs Fellow, AVAYA, USA
- Suzana Dragicevic, Simon Fraser University, Canada
- Dumitru Roman, Semantic Technology Institute Innsbruck, Austria
- Emmanuel Stefanakis, Harokopio University, Greece

Semantic Processing

- Marsal Gavalda, Nexidia Inc.-Atlanta, USA & CUIMPB-Barcelona, Spain
- Christian F. Hempelmann, RiverGlass Inc. - Champaign & Purdue University - West Lafayette, USA
- Josef Noll, ConnectedLife@UNIK / UiO- Kjeller, Norway
- Massimo Paolucci, DOCOMO Communications Laboratories Europe GmbH – Munich, Germany
- Tassilo Pellegrini, Semantic Web Company, Austria
- Antonio Maria Rinaldi, Universita di Napoli Federico II - Napoli Italy
- Dumitru Roman, University of Innsbruck, Austria
- Umberto Straccia, ISTI – CNR, Italy
- Rene Witte, Concordia University, Canada
- Peter Yeh, Accenture Technology Labs, USA
- Filip Zavoral, Charles University in Prague, Czech Republic

International Journal on Advances in Intelligent Systems
Volume 3, Numbers 1 & 2, 2010

CONTENTS

International Journal on Advances in Intelligent Systems, vol 3 no 1 & 2, year 2010, http://www.iariajournals.org/intelligent_systems/

Community Map Generation based on Trace-Collection for GNSS Outdoor and RF-based Indoor Localization Applications

Brian Niehöfer *, Andreas Lewandowski *, Ralf Burda *, Christian Wietfeld *, Franziskus Bauer† and Oliver Lüert‡
*Communication Networks Institute
Dortmund, University of Technology, Germany
{brian.niehoefer, andreas.lewandowski, ralf.burda, christian.wietfeld}@tu-dortmund.de
†Ford of Europe Product Development, Cologne, Germany
fbauer6@ford.com
‡mimoOn GmbH, Duisburg, Germany
oliver.luert@mimoon.de

***Abstract*—The paper describes a map generation system, which relies on random individual contributions of GNSS (Global Navigation Satellite Systems) traced movements for outdoor scenarios and traces from a RF-based indoor localization system. In a typical use case, mobile phone users would join a specific community to contribute their movements along streets, roads or pathways in form of so called journeys. The proposed algorithms presented in this paper are also able to generate a precise map for indoor scenarios, with which certain changes the accuracy of the RF localization system can be mitigated. Neither contributing subscribers nor the map generation need to have an á priori knowledge of the charted area. The approach presented here comprises the journey recording, the upload process to a common server and the processing algorithm for map generation. The filter mechanisms and adaptive plausibility checks applied to the raw data are key components for the deduction of precise street and movement maps. Summing up, the paper provides a proof of concept of a map generation system for different in- and outdoor application scenarios relying on random individual movements to demonstrate the performance of the algorithmic approach and the adaptablity of the map generator.**

***Keywords*-map generation; community map; street-attribute creation; RF-based indoor localization.**

I. Introduction

Recently GNSS (Global Navigation Satellite System) receivers have become a standard feature of business class mobile phones. Thus, a new and rapidly growing user group is employing terrestrial navigation and merges with the already proliferating market of vehicle based and portable navigation systems [7][8]. Assessing system performance by traits as ”being up to date”, ”good accuracy” and ”availability of annotations”, traditional GPS navigation systems always rely on already out–dated map data and thus are always lagging behind. Moreover they are limited in their resolution by local map storage capacity. In contrast community based map collection and generation scales with the number of members and is envisioned to overcome these limitations. In addition, memory demand on the mobile user side may be lowered as the used and locally stored map area at some time only need to be a subset of all available maps at the expense of increasing data volumes being exchanged. The dynamic upload of requested map data implicitly enables the use of up to date information. From the market perspective, such an approach must ensure an additional value for potential users in order to be accepted, e.g., in terms of accuracy, pricing or available annotations to points of interest [3][4]. Potential user groups range from pedestrians, joggers or hikers to car drivers. The performance indicators, as mentioned above, are assumed to match this requirement and in addition enhance coverage (i.e., in rural and retired areas). As GPS or other navigation resources such as GALILEO [13] are expected to be integrated into even low budget devices soon, the potential number of subscribers to a community based system is likely to rise. The minimum number of potential, technology enabling subscribers, therefore is supposed to be already active. Thus the authors strongly assume that an operational system, which follows the algorithms proposed in this work can be established in the current market situation.

Another emerging technology field is the application of RF-based indoor localization systems (e.g., in safety critical scenarios). In comparison to [1], this paper shows further extensions to the map generation algorithm addressing the specific conditions of an indoor positioning system. In comparison to GNSS systems, RF-based localization systems are highly vulnerable to changes in the environment caused by moving persons or any relocations of inventory. RF indoor localization relies on the evaluation of the Received Signal Strength Indicator (RSSI), a Time of Arrival (ToA) measurement or hybrid approaches. As typical application areas are highly influenced by multipath fading effects, the accuracy of position estimation is varying over the time. Considering a high number of journeys collected in these environments, temporal signal fluctuations caused by specific conditions can be mitigated using the map generation framework. Once a sophisticated user map is generated, the position estimation can be supported by the generated map during normal operations. In case of changes in the traffic infrastructure, new journeys are detected by persitent deviation and consequently lead to map updates. This process is envisioned to also support the detection of temporal situations like blockings due to accidents or construction sites.

Parameters of the depicted algorithm used in this paper have been deduced by pure heuristics. It should be noted that the goal of this work has been to prove the concept of a reasonable solution for distributed map processing rather than finding an optimum. The authors want to emphasize that the key contribution of this paper is the architecture of the novel, community based approach, especially for indoor scenarios. Furthermore, performance evaluations for motorway, city and pedestrian scenarios for GNSS journeys are shown.

In the sequel Section II describes the overall system to record journeys and to transfer them to a server. The section also addresses the processing stages of map merge and map storage. Section III gives details on the particular map generation algorithms while pointing out quantified indicators for proper map performance. Section IV describes the scenarios, which have been applied for performance evaluation. Section V provides results, explaining exemplary maps and presents quantitative measures. It is assumed that an increased size of the set of journeys including a particular route is sufficient to guarantee convergence of the traced data to a street map of reasonable quality. Section VI wraps up the major findings in a conclusion and states an outlook to further work.

II. System Overview

The architecture of the proposed map generation system is flat and operates in a client server fashion. It is based on a large number of mobile *trace recording devices (TRDs)*, e.g., a group of *collaborative navigation devices (CND)* or, in case of indoor scenarios, corresponding *indoor localization tags (ILTs)*. Furthermore one *central server or map generation server (MGS)* is necessary to accomplish the map generation process. In the first place, the MGS is responsible for filtering the raw traced data as well as for generating and storing the resulting map. In addition, it will keep track of a limited trace history for plausibility checks and finally hadles the task of scheduling the distribution of requested results back to the CNDs. It should be noted that in a practical use case TRD and CND are likely to be co–located or even different functions of the same physical device, although this is not a prerequisite for successful operation.

Figure 1 provides an overview of the community based map generation system. It shows the graphical user interface of a sample collaborative navigation device (community user) and the server providing databases for the recorded journeys and the resulting map. A further task of the server is the cumulation of the trace data into a concise map, which can be returned to the systems users.

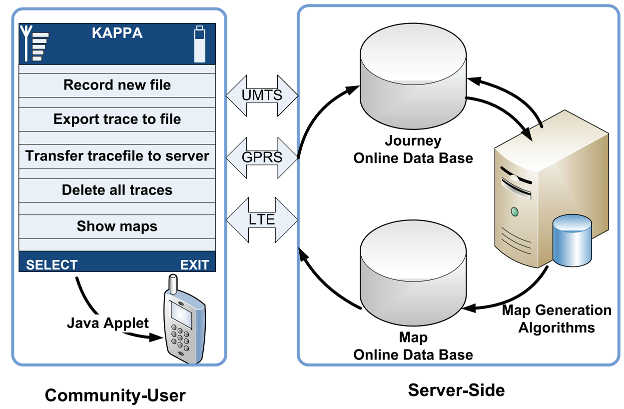


Figure 1. System Overview

A. Collaborative Navigation Device (CND)

The used CND is a simple GPS equipped mobile phone, which has been used as a trace recorder. It executes a Java-Midlet and stores the recorded waypoints as list of journeys (see Figure 2). Each journey is defined by an arbitrary

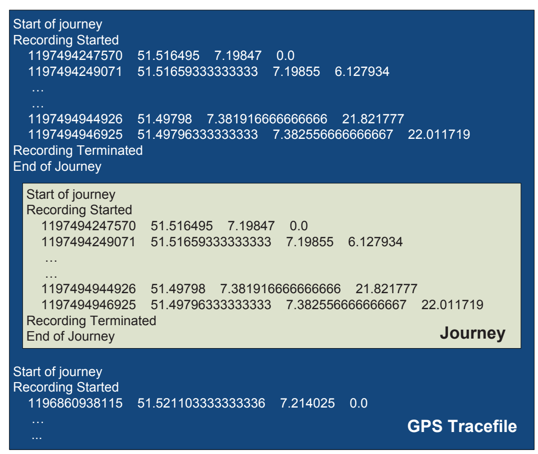


Figure 2. Sample Trace Data

number (>= 2) of waypoint samples. The first and the last sample denot the starting point of the journey and the ending point respectively. For the findings in this paper an initial setup has been chosen, which samples waypoints and records them in time intervals of 1 second. Each waypoint summarizes the geographical position in latitude and longitude along with an UTC time stamp. Altitude information is omitted due to the poor GPS resolution in this respect. Further annotation of the location date is possible in general but has been omitted for clarity as it would not support the concepts focused here. Figure 2 shows an exemplary part of a GPS trace file, whereby the first number in each row resembles the time stamp, the second and third the latitude respective longitude position and the forth the calculated velocity for the past two waypoints. We calculate the velocity during the recording of the GPS data to save computing time at the server side. Note that each GPS Trace File is defined as a container only and is used to transmit the relevant data from the mobile to the MGS. For this reason those text files are always created after each upload request of the CND.

The TRD is different in the reference design for the indoor localization system. Here the data is captured and analyzed by a central localization server, whereby an ILT sends the current position directly over the air to the server. This setup has been chosen due to the limited resources of processing capabilty and battery lifetime in the TRDs.

B. Map Storage Architecture

Figure 3 depicts the *object flow chart (OFC)* diagram for all data objects from raw journey data, summarized in GPS Trace Files as explained in II-A, to a final map knowledge representation. To facilitate the association of journey data to the final map, another storage entity is introduced, which will be called a *segment* throughout this paper. Each segment consists of a processed version of the former waypoints. In a first step, all recorded waypoints have to pass at least three redundancy checks, to minimize the effect of possible erroneous measurements in the GPS tracking process, before the connecting edges are accepted as a segment. Afterwards each segment is expanded by some additional information like an unique ID or the average velocity on a specified segment part, but this will be explained in Section III-D in detail. All segments are assumed to represent a potential path in the final map. The differences between a segment S and a journey J are the eliminated illogical waypoints W by using Consistency Checks. So (S) is an annotated subset of $\bigcup J$. (This will be will be explained in more detail in Section III-A. Each segment $S_y = \{N_1, N_2, \cdots N_n\}$ is made up by an ordered sequence of nodes N. The set of segments $\mathbb{S}$ comprises all nodes:

$$\bigcap_x S_x = \mathbb{N} \quad \text{with} \quad S_x \in \mathbb{S} \tag{1}$$

finally constitutes the points, which are known to the map. The connections between 2 or more segments are also special nodes called a *Crossing* (C), which are defined by

$$\exists(x, y) \rightarrow \mathrm{C} \in S_x \wedge \mathrm{C} \in S_y. \tag{2}$$

The relation $\mathbb{C} \subseteq \mathbb{N}$ is self evident. Each crossing C in turn may be entered via so called incoming segments $\mathbb{I}_C \subset \mathbb{S}$ and may be left via exiting segments $\mathbb{E}_C \subset \mathbb{S}$. In the general case $\mathbb{E}_C \bigcup \mathbb{I}_C = \varnothing$ is allowed. Crossings and the respective input and output subsets are stored in a so called Segment-Transition-Table.

The mentioned segment attributes are relevant for the plausibility check and intelligent filters. Furthermore, a source table stores the history of a segment with the underlying journeys, which were used to retrieve the particular

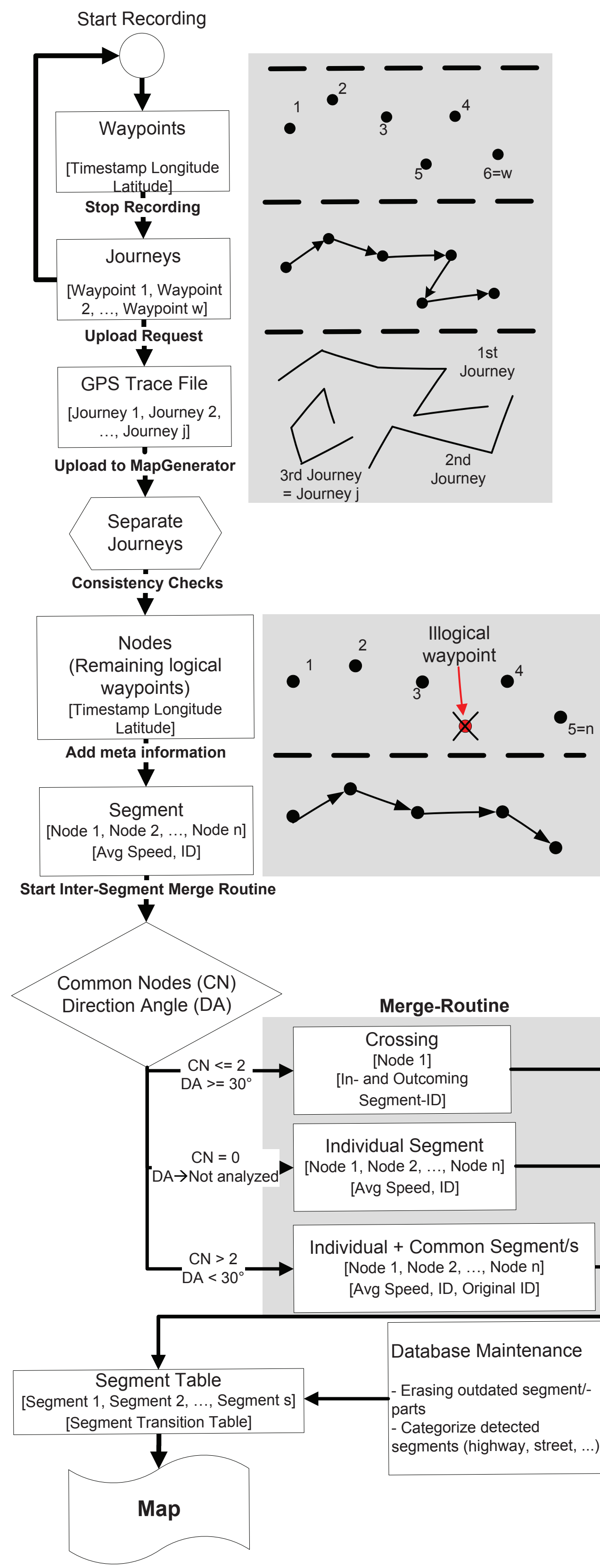


Figure 3. Object flow chart(OFC) diagram for map knowledge

segment. This enables an a-posteriori correction of the generated map in case subscribers or individual journeys turn out not to be trustworthy.

III. MAP GENERATION ALGORITHMS

In a first step, we collect journeys from different subscribers and analyze them. In order to retrieve a realistic map constellation, it is also necessary to analyze every trace for potential duplicates of journeys, say the way to work on different days. In general all journey data is subject to noise due to positioning inaccuracy. The major task of the pre-filter for the map generation is the detection and elimination of duplicates by means of an intelligent reasoning in order to yield a realistic graphical display.These filter algorithms are described in the following.

Presently, all filter algorithms are operating at the server side, although it would be possible to export some to the TRD, like the upcoming *Intra Journey Merge Routine (IJMR)*. To keep the algorithm-structure clearly and transparent, we decide to defer this aim up to the end in our implementation plan for the future.

Currently all waypoints and recorded journeys may contain evident errors caused by measurement errors from the CND. Therefore, the GPS annotated journeys are checked for inconsistencies. Second, each journey is investigated for multiple passings of the same street. Third the journeys are compared to the existing map data (created by merging already uploaded journeys), the distances are calculated and, where appropriate, journey data is merged with the map data. Finally, the information in the journey data is processed to add additional information to the segment- and the transition-table of the map. Note that all the assumptions for validity and the thresholds for taking actions on the journey data are purely heuristic at this stage of research. They have been retrieved and validated to be useful by a lot of experiments.

A. Consistency Checks

As a first step of validation each trace is checked for obvious inconsistencies. This check starts with a pre-run, which executes a 'Minimum distance between points check'.

Due to the inaccuracy of public GPS or RF-based indoor localization, all waypoints, which are too close ($|W_i - W_{i+1}| < \mathrm{T}[\mathrm{m}]$) to a given intermediate starting point are averaged and replaced by the result. T is assumed to be a valuable threshold for different application scenario and localization system. In outdoor scenarios, the averaging process is stopped e.g., when

$$\frac{1}{w}\sum_{l=1}^{w}|W_l - W_{w+1}| \geq 5[\mathrm{m}]. \tag{3}$$

Once this spatial noise filter has been applied, the remaining annotated journey data is subject to three further checks (exemplary parameterized for a motorway scenario) .

1) **Speed:** The speed calculated from time and spatial distance between two recorded points may not be larger than $200\frac{km}{h}$. If this is the case, the second point is assumed to be faulty and discarded.
2) **Acceleration:** Similarly, the acceleration along a journey is checked. If the acceleration is larger than $4\frac{m}{s^2}$, the point, which was reached through this unlikely acceleration is discarded.
3) **Direction change:** The angle between two successive segment fragment vectors may not be higher than a certain speed-dependent angle. If this is witnessed in the journey the end point of the second vector is discarded.

The filter parameters are adapted to the use case scenario, as different mobility patterns for indoor and outdoor scenarios are expected. If a waypoint is marked by one of these consistency checks, it is discarded from the journey to smoothen the resulting way. The check is then re-run until no further eliminations have been executed. An example for the direction plausibility test is given Figure 4.

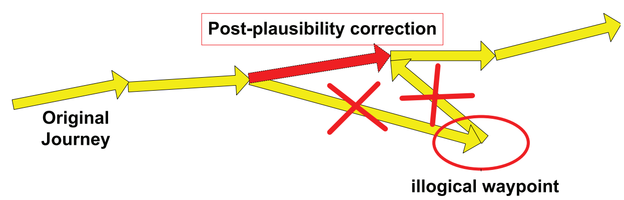


Figure 4. Effects of maximum angle check

The spatial points, which remain in the filtered output of this processing are now called *nodes* N. Due to the averaging process and the other checks $\exists i \rightarrow N_i \ni \mathbb{W}$ is possible. Note that $|\mathbb{W}| \geq |\mathbb{N}|$ holds true.

B. Intra Journey Merge Routine (IJMR)

The upcoming merge routine only controls complete segments. Therefore, redundancy has to be eliminated by checking each journey for intra merge parts. Thus, only relevant waypoints are passed to the segment processing on the server. Redundancy is assumed when a subscriber drives the same street, or a part of it, multiple times per journey, e.g., while searching for a parking lot. The merge routine compares the distance between each fifth (chosen arbitrarily to return processing load) segment fragment of the journey. When the distance decreases below a certain threshold, the routine cuts this part and replaces it by artificial but concise waypoints attributed by new index numbers, directions of travel and time stamps. The processing is then repeated. If none of the 3 consistency criteria is violated, the program passes the result to the segment generation and continues to analyze the next journey. The dissection of the journey into multiple segments enables the reuse of the algorithm for the upcoming *multi journey merge routine (MJMR)*. Both merge

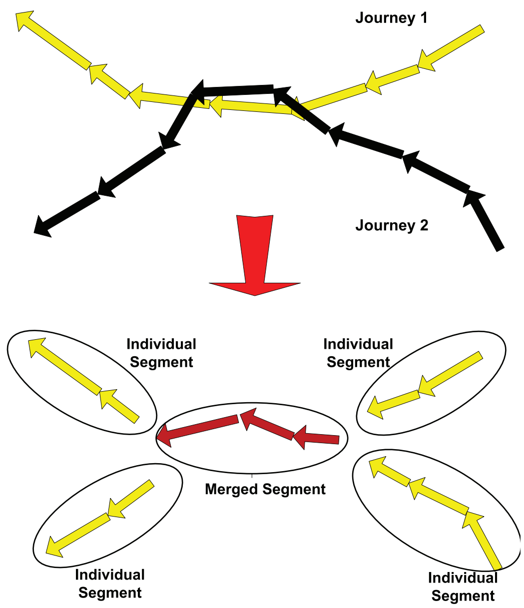


Figure 5. Trace split/merge to segments

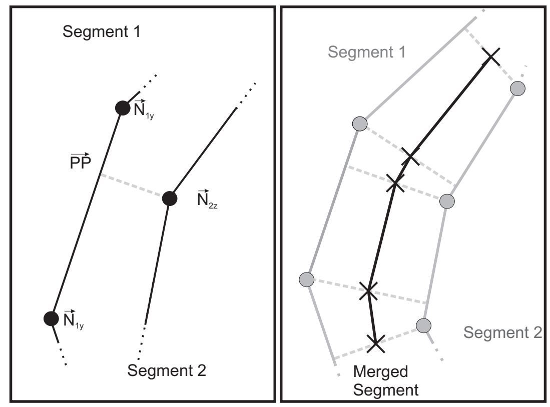


Figure 6. Perpendicular segment alignment

steps are, at present, executed on the server to maintain a concise processing engine. Deployment of the IJMR to the CND or ILT results in a proportional scaling of processing power and journey count in a productive system.

C. Perpendicular Calculation

In a first attempt we compare the distance of all nodes of a new segment record to all nodes already stored in the database. In case of being under a given threshold T (e.g., 13m for GPS outdoor scenarios), the segments are supposed to be copies of already known ones. Moreover, distances between a node and the connecting line between close neighbors may meet the same criterion as being shown in Figure 6-left. To include this case in our algorithm, we use a point–to–point–distance calculation to the closest neighbor node and the line–to–point distance between the predecessor and successor. Now it is possible to calculate the minimum distance to a line, by using the formula displayed below.

$$\left(\vec{N_1} - \vec{N_3} - x \cdot (\vec{N_2} - \vec{N_1})\right) \cdot (\vec{N_2} - \vec{N_1}) = 0 \tag{4}$$

$$\vec{N} = \vec{N_1} + x \cdot (\vec{N_2} - \vec{N_1}) \tag{5}$$

By calculating the 'x' value for which the dot product becomes zero, we draw the perpendicular from the $\vec{N_3}$ in the detail pane of Figure 6-right to the track between $\vec{N_1}$ and $\vec{N_2}$. Obviously, this is the minimum distance between the point of one journey and the segment fragment of the other. Now we can decide precisely, which segments or parts of them (fragments) are below T and feed them to the merge process. Each perpendicular minimum will then be used to increase the reliability factor α of a merged waypoint, which can be understood as an attribute of 'trust' in the correctness of this point. Above a particular numerical threshold of α a segment (either genuine or merged) is consolidated into the final map. For the further calucation steps no distinction is made with respect to the origin of a (now qualified) segment. The weight of a segment in the map is monitored constantly with reference to the count of supporting decisions from new segments or their fragments. The reliability is used as a priority factor in further calculations, like shown in Formula 6, where α is the reliability factor.

$$\vec{N}_{final} = \frac{\alpha_1 \vec{N}_{new} + \alpha_2 \vec{N}_{old}}{\alpha_1 + \alpha_2} \tag{6}$$

In case of a new segment, the reliability factor $\alpha_1 = 1$. Superior segment priority invalidates new *close by* segment fragments, which are still too close to make up for a new segment or street. Aged (not reconfirmed) segments are subject to ageing and consequently will be deleted by a database maintenance routine eventually. Merging two segments of equal α in this perpendicular way leads up to twice the number of estimated nodes than pure averaging as shown in the right pane of Figure 6. The increased node density enables a smooth fitting even to curved trajectories and thus is more realistic. The outer circles in Figure 6-right account for the nodes of the two segment fragments, which we are going to merge in order to eliminate the inevitable noise in position estimation. Concerning our perpendicular calculation (inner lines and circles), we are able to reach highly concise results, as shown by the boundary lines.

By doing so, Figure 7 shows an iterative evolution of the community map (CM) for four exemplary journeys, whereby the dashed lines denote the new added one.

D. Street segment attributes

After inserting new or merged segments into the map, non spatial information is extracted from the segment. Obvious information is the average and maximum velocity (AV, MV) measured across a particular segment fragment and therefore

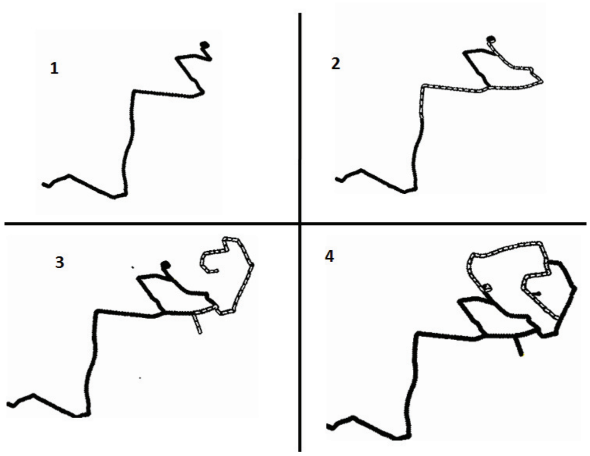


Figure 7. Iterative map evolution from 4 journeys

across the segment it has been associated to. AV then differentiates between highways, streets and walkways in our model, based on characteristics like average speed, direction changes, etc.. Such an approach is useful to deduce different streets, which are directly in parallel, e.g., exit or entry lanes on highways. Further attributes like the time-of-day or day-in-week including the respective AV and MV values per hour are already stored but still remain unused in the algorithms. This information for post analysis is valuable in case the optimization goal of navigation systems employing the CM is minimum delay rather than minimum distance. Community users may update all these attributes by real-time measurements. Taking into account age and first order derivate in case of changing attributes, navigation software gains superior capabilities to detect temporal changes in traffic situations (road work, traffic jam, detours) in nearly real–time.

E. Extensions for new application scenarios

For applying the map generator to indoor scenarios, several changes are implemented to adapt the changed mobility patterns and the different positioning accuracies. The consistency checks have been adapted for the following *movement-classes*:

1) High speed (outdoor)
2) Vehicular (outdoor)
3) Low speed (outdoor)
4) Ultra low speed (indoor)

The degree of freedom for the movement paths is enlarged with every step by reducing the strictness of rules. In the special case of indoor scenarios, we implemented a double stage process, as high fluctuations in the RF-based localization system are observed:

1) Erasing outliers with a coarse filter
2) Adaptation of remaining illogical waypoints by perpendicular calculation

The merge routines have been adapted in terms of dimension and degree of freedom. Additionally, we introduced a speed depended smoothing of the estimated positions.

IV. Scenarios for Performance Evaluation

In order to show the adaptivity and performance of the map generator framework for different scenarios, we chose a top-down approach in terms of scenario size. At first, a classic motorway scenario is evaluated, before showing a city traffic scenario. These are the classic application scenarios for today's navigation devices. But hence, pedestrian mobility in combination with navigation is opening new application fields. Therefore, we provide two scenarios. One outdoor scenario, where the position estimation is accomplished by GNSS, and one indoor scenario, where the position estimation is accomplished by a RF-based localization system.

A. Motorway Scenario

The journeys for the classical Motorway Scenario have been generated in Bochum, Germany (GPS-coordinates approximately: 51.516 N, 7.206 E) at a motorway cross of the frequently driven A40 and A43. Average speeds of around 80 km/h are expected, as the area is mostly speed limited due to heavy commuter traffic. The street structure is characterized by wide spaces and a straight street course.

B. City Traffic

City Traffic is different in terms of speed and position changes compared to motorway scenarios. We have generated GPS journeys again in Bochum. The average speeds are expected to be at around 30 km/h. An interesting point will be how reliable the map generation is working in that application field, as environmental influences like high buildings affect the accuracy of GPS positioning. As the structure of city streets is more fine grained compared to motorways, it is more challenging to generate a highly accurate CM.

C. Pedestrian Movement

In order to evaluate the accuracy of the map generator in pedestrian scenarios, we traced several journeys on our campus on the classical route from out institute to the refectory (Mensa). The journey takes about 3 minutes over a distance of 350m. This scenario is challenging as surrounding buildings affect the accuracy of GNSS position estimation. We deployed an additional tracking application on the iPhone as CND, as we need fast acquisition of the initial user position. In our case this has been accomplished by utilizing the cellular network information to improve the startup performance of the used GNSS system (GPS). This scenario has been chosen to show the capability of adaptation of the map generator framework for low speed and dense position data.

D. Indoor Localization Testbed

RF-based indoor localization is a challenging research topic, as the radio channel is prone to interferences caused by fixed infrastructure, other mobile users and the CMD itself. Hence, the position estimation is highly susceptible to these effects, when relying on the common techniques like Received Signal Strength Indicator (RSSI) or Time of Arrival (ToA) range estimations. When thinking of a target environment like an industrial application scenario, employees are expected to walk over commonly used movement paths, as the degree of freedom is usually restricted by constructional circumstances. Thus, the map generator approach constricts the parameter space of journeys by a system calibration time, where external influences are strictly avoided. Hence, during normal operation, external influences are mitigated as the position estimation is corrected to the map location.

In the context of this paper, it is very interesting to learn about the scalability of the map generator in terms of size of scenario and movement behavior. We analyse a mobile indoor localization testbed, consisting of a model train as depicted in Figure 8.

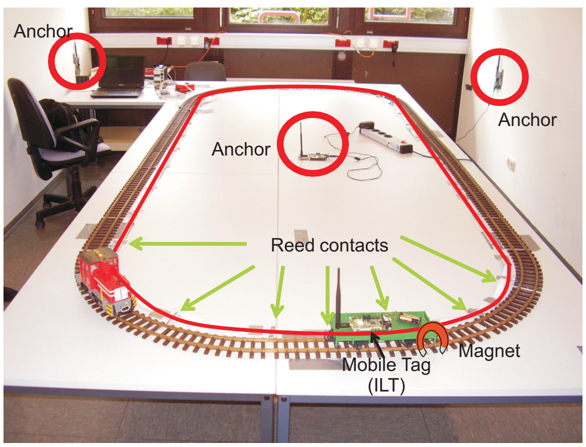


Figure 8. Indoor localization testbed for performance evaluation of the map generator

A so called Chirp Spread Spectrum (CSS) [12] localization system is applied, which uses ToA for range estimation. This system is an extension of the well known IEEE802.15.4 standard [9] and is standardized in the extension IEEE802.15.4a [10] as an alternative Physical Layer (PHY) implementation. We installed 5 fixed anchor nodes and placed an ILT on the model train. The localization error is expected to be at a maximum of 2 meters using this constellation. This is obviously too high for such a small scale scenario. Therefore, the generation of a movement map could, in this case, enhance the localization accuracy and reduce the number of position measurements, as the position change can be predicted based on the current speed of the user.

V. Results of the Performance Evaluation

The following chapter describes an analysis of the convergence of the street segment representation. All segments have been derived as result of the merge from several journeys for the described application scenarios.

A. Motorway

The two highway sections shown in Figure 9 are the A43 and the A40 crossing in the bottom right at Bochum junction (GPS-coordinates approximately: 51.516 N, 7.206 E). Figure 9 depicts a map and a corresponding Google Maps screenshot in an increased diameter. Recorded average speeds below 80 km/h are indicated by a dashed, faster segments by a solid line type. The grey circles mark deduced crossings between at least two map segments. The gap in

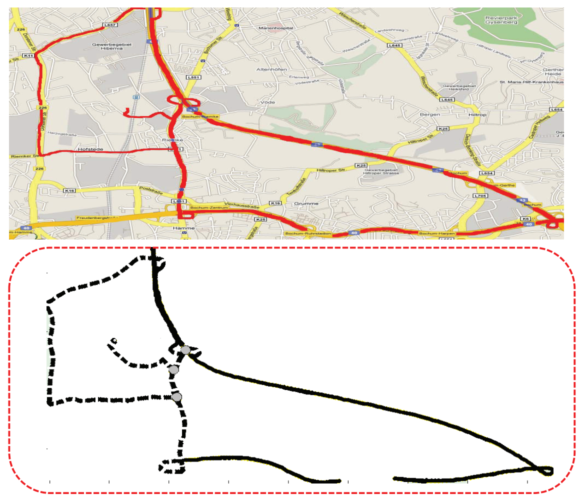


Figure 9. Reference check of deducted map

the A40 segment results from a tunnel between the exits Bochum-Harpen and Bochum-Ruhrstadion. To clarify the original route and the reached accuracy, Figure 9 (top) shows a Google Maps screenshot in an overlay with the driven route.

This vivid example clarifies the achieved approvements. In contradistinction to the Google Maps data origin, the map generator must not work with specially equipped time- and cost-intensive measurement cars. Concerning the fact, that our map generator just uses a high quantity of CND position tracks (e.g., from mobile phones), we achieve a very prompt overall map, demonstrating an accuracy which is competitive to that of Google.

Figure 10 depicts the average error of a map segment related to the underlying journey count. At this point, the authors want to mention, that the used journeys were tracked with an external GPS antenna, supporting Differential GPS (DGPS). As reference we used the entire street width. Hence, a deviation ΔD of 0.20m does not correspond with an accurate localization (compare with Figure 10), it just figures out how accurately and how fast the map generator is able to create a realistic street position and -course. Based on

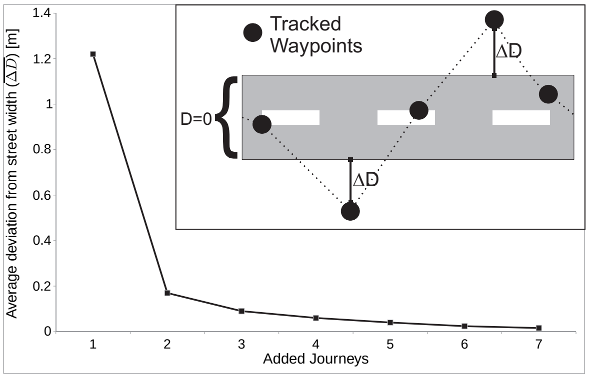


Figure 10. Mean deviation from roadside vs. journey count

this error measurement, we assume that merging 4 to 6 segments describing the same street can result in a sufficient representation ($\overline{\Delta \mathrm{D}} \leq 0.1\mathrm{m}$) of a street segment existing in reality.

B. City Traffic

Figure 11 shows the evolution of the map generation process for the evaluated city scenario. It can be seen that the resulting map perfectly matches to the Google Maps view. The evolutionary map generation process is depicted in three major steps, following additional journeys taken into account for the computation.

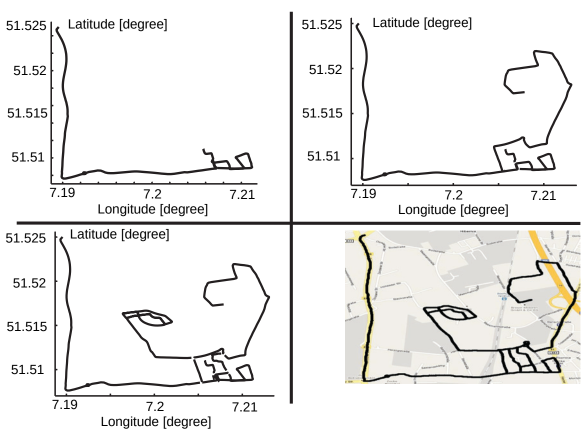


Figure 11. Map Generation for the City Scenario

For a productive service the effect of convergence and thus the effort up to an initial map generation is important. Claiming that a map is ready for use when new data has not caused major changes in the database for some journey additions, we focus on the quantity of information particular journeys contribute, named $I(J)$.Therefore, we analyse after every merge process, how many new nodes have been contributed by the actually beholded journey to the overall map. In case of a complete new segment, every node carries information about a street-part, which was not detected yet ($I = 1$). On the other hand, it is also possible, that some journeys describing already detected parts of the road network, so the information content of such journeys is $0 \leq I(J) < 1$.

In general these results will depend on many factors such as the quality of available journey data, too. However, this test of the processing gives a good insight into the common map generation.

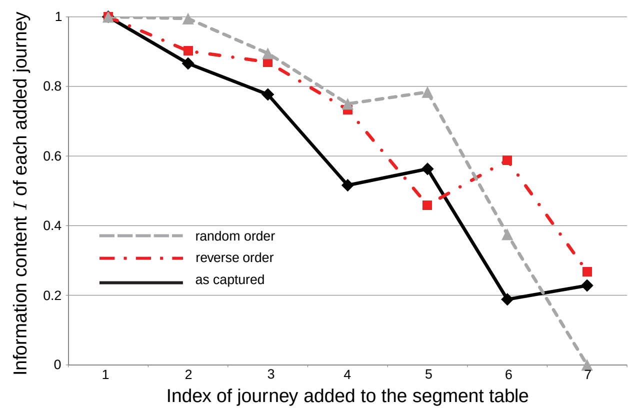


Figure 12. Information content $I(J)$ of each journey added in three different orders

Figure 12 depicts the results for three different processing orders of 7 journeys on the paths shown in Figure 11. The area covered is of about $0.5\mathrm{km}^2$. It can be seen that the information content of a new journey is depending on the number of already analyzed journeys. Beneath the order, other parameters like position and length of journey, do also have a high influence on the information content. A short journey on a main road do not have a hugh information content to an already created segment table, whereas a long journey, which just uses small side roads, can depict a very hugh information content, even when merging such a journey to an already existing map.

These effects are also visible in Figure 12. In every order the first added journey shows an information content of $I = 1$ and the fifth journey of the random order, for example, has a higher amount of information content than the forth one, though it was added afterwards.

It has to be noted that this test is only a first hint and that the authors are aware of the fact, that the absolute amount of information in a journey will show significant variations depending on a multitude of parameters such as length of journey, precision of data, novelty of taken path etc..

C. Pedestrian Movement

In order to demonstrate the accuracy, scalability and behavior of the map generation framework for lower speeds and therefore a geographically dense set of waypoints, Figure 13 depicts the result of a journey collection and map generation process on our campus. The journeys have been

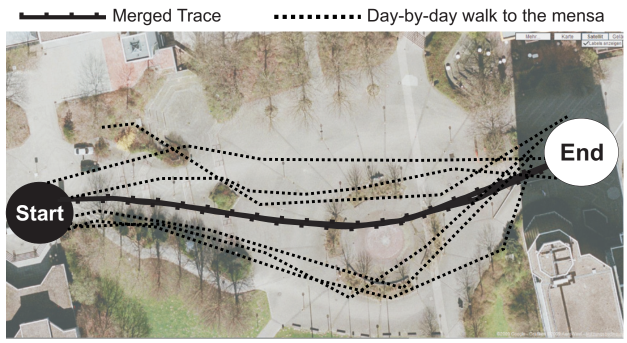


Figure 13. Merged Segment in comparison to measurements

collected by a pedestrian user walking different ways to the refectory. If a map for this scenario is generated, the merged journeys lead to a segment located in the middle of the journeys, as expected. It is also visible that outliers in the measurement or movement pattern are mitigated by the number of journeys taken into account. Again, the resulting map is depending on the accuracy of position estimation, which obviously is high in this scenario, as different movement paths can be detected clearly. This benefit is gained by the combined localization process and the sophisticated GNSS receiver of the iPhone.

D. Indoor Localization Testbed

This experiment again give an insight into the performance of the filtering and merging algorithms as being described in Section III. Figure 14 depicts the preliminary results of the journey collection without applying the algorithms of the map generator. The position deviation from the train lane is significant and obvious. Furthermore, it is fluctuating for each circular driving under static conditions. We have interconnected three independent cycles, which explains the starting and ending points of the journeys. As depicted in the figure, the RF-based indoor localization system shows position dependent fluctuations in the accuracy of position estimation. The overall performance of the system shows a maximum error of around 1m, where in most cases the accuracy is considerably better.

Taking into account journey 3 from Figure 14, Figure 15 shows the result of the consistency checks alone before the merging to the final map is executed. It can be seen that outlier measurements are mitigated. The consistency checks have been performed using the following parameters:

1) Speed Limit: $1.5\frac{m}{s}$
2) Acceleration Limit: $0.5\frac{m}{s^2}$
3) Vector Plausibility: $40°$

In the next step, we executed the intra journey merge routine in order to generate a more accurate map of the

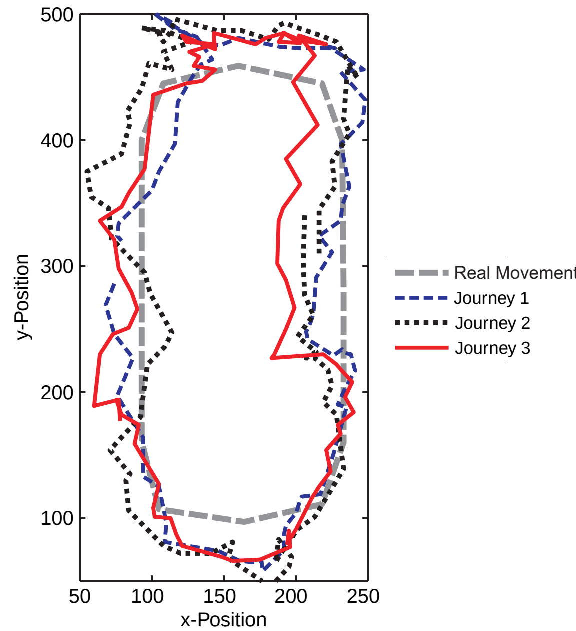


Figure 14. Comparison of measured journeys (raw-material) and the real movement

scenario. In this case we merged three journeys after the plausibility checks. In Figure 16 it can be seen that the map has closely adapted to the real train lane. It is also visible that constant fluctuations of the RF localization affect the accuracy of the map. As a plus, these adverse effects seem to be confined to the area of their incident. In comparison to the raw data, the generated map greatly enhances the accuracy of the localization process. Summarizing, the accuracy of the map reduces the maximum error by 0.25m in this case.

In the final system design, the map generation algorithm can be used to generate an up to date map of the application scenario. When thinking of an industrial or public indoor environment, the available maps are as old as the building itself. Constructional changes are often not reflected in building plans. Hence, these blueprints are not usable for optimizing the performance of an indoor localization system.

After generating the map, the user position can be estimated by a combination of CM and localization system by applying a particle filter approach [11]. The positions estimated by the system are represented in so called particles, which are weighted and mapped on the new building map. By doing so, outliers can be mitigated during the operational phase of the system, but hence the degree of freedom is reduced. Concluding, a certain update interval for the user map is mandatory to retain the accuracy of localization.

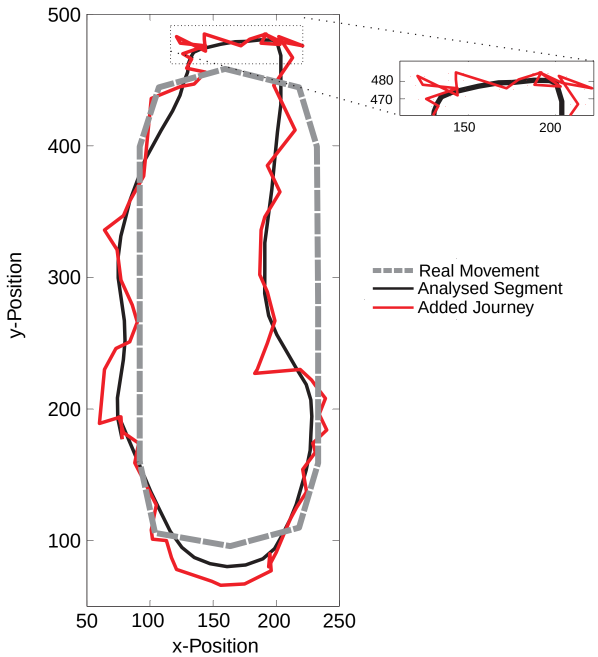


Figure 15. Raw Material before and after consistency checks

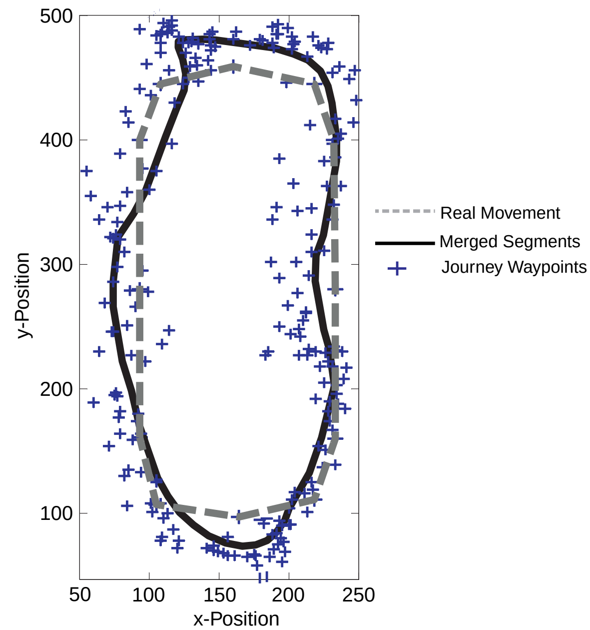


Figure 16. Merged Segment in comparison to measurements

VI. Conclusions

This paper describes a successful experimental implementation of map generation out of GPS journeys recorded by several mobile subscribers. Employing attribute tables for each street segment should enable an enhanced navigation service. The combination of the Java Applet and the Generator Server Program allows the user to create map data plus lots of other useful information for navigation. The initial results of these experiments suggest that only 4 to 6 journeys along a particular street segment result in a useful estimation even in complicated scenarios. Furthermore, we have demonstrated the scalability of the proposed map generation framework by exploring four scenario classes. The results demonstrate the wide applicability of the approach. Especially indoor scenarios can benefit from this framework. The mean position deviation has been reduced considerable.

These promising findings motivate further extensions, including 3D-Navigation by means of additional sensor data (like accelerometers) or optimizations for noise reduction and increased accuracy. Power awareness for long battery lifetimes and communication protocol dependent topics like traffic models or business case schemes will be studies in upcoming work. Additional features as described in [6] will enhance the controllability of the consumer hardware. The influence of the sample interval to the fidelity of the resulting map has to be carefully pondered against the resulting workloads (in terms of bandwidth usage, storage requirements etc.) in the system and on the radio link.

VII. Future Work

The map generation framework is capable for opening new application fields for traffic management and indoor localization applications.

The next step will be that the map generator algorithms are optimized for real-time calculation. If this target is reached, today's traffic forecast procedures can even benefit, if e.g., working sites on motorways can be detected out of the transmitted journeys. This is a realistic target, as the GNSS accuracy will be enhanced with the launch auf Galileo. Furthermore, if a temporal connection between navigation devices and traffic management center can be established e.g., by using public cellular or satellite networks, journeys and route planning can be transmitted on the uplink and be regarded for traffic jam prognosis; updated traffic information for adaptive routing and lane assistance can therefore be forwarded to the navigation devices of the user as a special service of the operator.

The applied consistency checks are very useful for indoor localization systems. If the calculation time is optimized, these algorithms can be applied in a local control center. As we have seen that the accuracy can be enhanced, this approach will lead to reduced costs, as the investment in sophisticated localization devices can be reduced.

Acknowledgment

This work is conducted within the SAVE Project (Geographic Information System for Alarming using autonomous, networked Gas Sensors) and is funded by the German Federal Ministry of Education and Research (BMBF) – 16SV3711

REFERENCES

[1] Niehöfer, B., Burda, R., Wietfeld, C., Bauer, F., and Lüert, O., GPS Community Map Generation for Enhanced Routing Methods based on Trace-Collection by Mobile Phones, Advances in Satellite and Space Communications, 2009 (SPACOMM), pages 156-161, Colmar, France, 2009

[2] Bruntrup, R. and Edelkamp, S., Incremental Map Generation with GPS Traces, Intelligent Transportation System, 2005. Proceedings. IEEE, pages 574-579, 2005

[3] Edelkamp, S. and Schroedl, S., Route Planning and Map Inference with Global Positioning Traces, Computer Science in Perspective, pages 128-151, Springer Verlag, 2003

[4] Kaasinen, E., User needs a Location-Aware Mobile Services, Personal and Ubiquitous Computing, Vol. 7, pages 70-79, Springer Verlag, 2003

[5] Duckham, M. and Kulik, L., Simplest Paths: Automated Route Selection for Navigation, Lecture Notes in Computer Science, Vol. 2825, pages 169-185, Springer Verlag, 2003

[6] Robbins, D. , Cutrell, E., Sarin, R., and Horvitz, E., ZoneZoom: Map Navigation for Smartphones with Recursive View Segmentation, AVI '04: Proceedings of the working conference on Advanced visual interfaces, pages 231-234, Gallipoli, Italy, 2004

[7] Schroedl, S., Rogers, S., and Wilson, C., Map Refinement from GPS Traces, Technical Report RTC 6/2000, DaimlerChrysler Research and Technology North America, 2000

[8] Zhao, Y., Standardization of Mobile Phone Positioning for 3G Systems, Communications Magazine, vol. 40, issue7, pages 108-116, IEEE, 2002

[9] IEEE 802.15.4 Workgroup, IEEE Standard for Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs), New York, NY, IEEE, 2003

[10] IEEE 802.15 Workgroup, IEEE 802.15.4a Standard, WPAN Low Rate Alternative PHY Task Group 4a: IEEE Standard for Wireless Personal Area Networks Low Rate Alternative Physical Layer (PHY), 2007

[11] Widyawan, M.K., Klepal, M., and Pesch, D., A Bayesian Approach for RF-Based Indoor Localisation, Wireless Communication Systems, 2007. ISWCS 2007. 4th International Symposium on, pages 133-137, Trondheim, Norway, 2007

[12] Kang, H., Seo, G.w., and Lee, J., Error Compensation for CSS-based Localization System, Proceedings of the World Congress on Engineering and Computer Science 2009 Vol II WCECS 2009, San Francisco, USA, 2009

[13] Enderle, W., Weber, T., and Kuhlen, H., The Architecture of the European Global Navigation Satellite System - GALILEO, 10th Australian International Aerospace Congress incorporating the 14th National Space Engineering Symposium, Brisbane, Australia, 2003

National Electronic Identity Management: The Challenge of a citizen-centric Approach beyond Technical Design

Stefan Strauß
Institute of Technology Assessment (ITA)
Austrian Academy of Sciences
Vienna, Austria
sstrauss@oeaw.ac.at

Georg Aichholzer
Institute of Technology Assessment (ITA)
Austrian Academy of Sciences
Vienna, Austria
aich@oeaw.ac.at

Abstract— **National governments across Europe are currently introducing electronic identity management systems for enhancing security and gathering more unified forms of authentication for online public services. A particular challenge of security system design is to cope with the suspense between security and usability. This is strongly reflecting in identity management where this suspense becomes very apparent. Thus, for the success of identity management systems a certain focus on user centricity is demanded. This paper analyzes the system in Austria with respect to important determinants of a citizen-centric identity management approach, deduced from security usability issues, interrelated with factors for user perception as provided by the Technology Acceptance Model. The result reveals a biased picture of user centricity with an essential need for a stronger consideration of user perception and the provision of additional benefits addressing a perceivable user value.**

Keywords - electronic identity management; user centricity; security usability; e-government; Austria; TAM

I. INTRODUCTION

With the achievement of higher stages of interaction in online services and the increasing number of electronic transactions in different domains of everyday life, identity management (IDM) more and more becomes a crucial challenge in the information society as most transactions require user authentication. This is especially the case in the field of e-government, where IDM plays a particular role. The maturity of online public services allows users not just to obtain information but also to conduct transactions with public administration completely online via single sign-on (SSO). Currently, there is a rather broad scope of different concepts and technologies for authentication procedures in online services, which makes IDM a tricky task to cope with, especially for end-users. Therefore a number of countries in Europe are introducing systems for electronic identity management (e-IDMS) in order to improve security in online services and to set-up more harmonized forms of identification and the corresponding procedures. As these approaches at least aim to unify national IDM for e-government, they have a special focus on their citizens as the primary user group. Hence, user centricity is an essential factor and a certain challenge in this context.

A part of this work has been presented at the CENTRIC 2009 conference and this paper is an extended version of our contribution [1]. It upgrades the findings in [1] by emphasizing on issues regarding security usability and ties them in with user perception as a key determinant for user centricity.

Higher levels of security, efficiency and cost-effectiveness of electronic communication and transactions are the major benefits that governments expect from a national e-IDMS, for the public administration itself as well as for citizens and businesses. Lips et al. postulate that e-IDM becomes “the sine qua non of successful e-government”, and highlight two perspectives that have been dominating up to now: “technical design” and “privacy advocacy” [2]. They argue for transcending the preoccupation with these essentially instrumental views towards analyzing the wider societal implications of this innovation and for paying greater attention to social design issues [3]. McKenzie et al. refer to challenges, policy dilemmas entailed by multiple goals and failures in past IDM approaches, which have spurred the debate on appropriate overall e-IDMS strategies [4]. A number of normative frameworks for e-IDMS that deal with user-centric aspects have been suggested such as the set of principles for security usability developed by Jøsang et al. [5], the Seven Laws of Identity from Cameron [6] or the findings of the PRIME project [7].

These initiatives underline the paradigm shift that IDM approaches are currently experiencing towards a stronger focus on the user. User-centric IDM is expected to provide an individual “full control of transactions involving her identity data” [8]. In terms of security systems, security usability evolved as a special research field and is an integral part of user centricity though with a strong focus on technical design issues. More recent approaches of user centricity go beyond technical design and highlight the importance of a stronger integration of further aspects: e.g., applying experiences and techniques of the field of human-computer interaction (HCI) such as psychological and social aspects of usability in order to encourage a design perspective that comprehends the user as a part of the system [9]. However, there are several different concepts and understandings of user centricity (cf. [8] [10]) and research on national e-IDMS in terms of this aspect has been neglected. IDM approaches in e-government have some major differences compared to private sector IDM, with other determining factors to incorporate [4]; above all, governments have to care for broader aims such as social inclusion and interoperability, and at the same time governments have coercive power, which may lower incentives to be responsive to citizen concerns. As these and similar aspects have not yet received adequate attention in research on user-centric design of a national e-IDMS, this paper aims to contribute to closing this gap. On the example of the Austrian system, the peculiarities of an e-IDMS are explained with regard to security usability in order to grasp challenging aspects of citizen centricity.

The paper is structured as follows: Section II depicts the research design of the analysis. Section III elucidates general aspects of user centricity and the relation between security and usability. In Section IV, relevant determinants for user-centric IDM approaches are suggested. This is followed by an explanation of the Austrian e-IDMS and its specific design in Section V. The pursuance of user centricity in the Austrian system is analyzed in Section VI, followed by explanations for the current situation and considerations about approaches to cope with major challenges (Section VII). Finally, in Section VIII, the results of this analysis are summarized and concluded.

II. RESEARCH DESIGN

This paper analyzes national identity management from a user-centric point of view and makes a contribution to develop suitable approaches for overcoming the challenges in this domain. The analysis is based on a case-study of the national e-IDMS in Austria as a top-ranking EU-country in terms of e-government [11]. This study is part of a larger comparative research project on the introduction of national identity management systems in selected European countries. The empirical investigation (conducted in 2008) was a combination of several methods: A comprehensive literature review, an analysis of research papers, official documents, expert statements, technical reports and specifications, face-to-face interviews with key decision-makers and stakeholders at different governmental levels that were involved in the innovation process; and practical tests of the e-IDMS. For the analysis of this paper, following research questions were identified:

- What are the significant aspects of user centricity for security systems and which role do they play for identity management particularly in the context of e-government? Starting with a brief introduction of user centricity in general, we focus on key parameters relevant for user-centric national IDM.
- What are the major characteristics of the national e-IDMS in Austria and how does it incorporate user/citizen-centric parameters and the user's perception of the system? After describing specific features and peculiarities of the Austrian system we analyze them with respect to parameters relevant for user centricity.
- How balanced is the interplay between the relevant determinants for user centricity in the Austrian e-IDMS and what are major challenges for avoiding a trade-off between security and usability? Based on findings of security usability combined with considerations about user perception shed light on the current situation regarding user centricity.

Our methodological approach draws upon theoretical conceptualizations of user centricity, security usability as well as technical concepts in the field of identity management. The four basic architectural models of e-IDMS – siloed, centralized, federated and user-centric identity systems (cf. [5] [8] [12]) – show how technical IDM concepts evolved towards a user-centric architecture. Hence, we include these models in our analysis.

As there are a lot of different views and conceptualizations of user centricity that do not provide a commonly accepted delineation of the concept or a universal set of criteria, our approach is also informed by key dimensions of user centricity identified as predominant in the relevant literature. As the aim of our paper is to analyze user-centric aspects exceeding technical design, the methodology also orientates on Davis' [13] already classical Technology Acceptance Model (TAM), which provides a suitable framework for better understanding of user perception as it allows to grasp relevant determinants and explanations for the users willingness to get involved with a new technology. In terms of security systems and identity management, this is of special interest as the ambivalent relationship between usability and security demands for a stronger consideration of user perception in this domain.

The TAM describes the interrelation between system characteristics, perceived usefulness, perceived ease of use (i.e., usability) and attitude for usage and actual usage behavior, i.e., the intention to use. Davis defines perceived usefulness as "the degree to which a person believes that using a particular system would enhance his or her job performance". This addresses the users' perceived level of potential improvement of workflows through the usage of ICT, i.e., which benefits users expect from system usage. Perceived ease of use (usability) is defined as "the degree to which a person believes that using a particular system would be free of effort". Thus it refers to the users' expectations of the systems usability and the efforts usage implies. These two factors form the users' intention to use a technology and have impact on the individual attitude for usage of a system and thus on the resulting usage and acceptance of the system itself [14] [15].

III. ASPECTS OF USER CENTRICITY

In general, user centricity can be described as the manifestation of a certain demand for more user-orientation in technology. Placing the user and his demands in the center of technology design should provide him more control and user value. The rapid technological progress and particularly the emergence of internet services played a certain role for the paradigm shift towards a user-centric view. The decentralized structure of the internet, distributed architectures and the increase of online service created further complexity and new user-requirements for the implementation of usable technologies and services in this context, with user centricity becoming an essential aspect. This reflects in many different domains and in particular in the field of e-government.

A. The Suspense between Usability and Security

In terms of security, user centricity is a particular challenge as it addresses the tense relation to usability. Zurko defined user-centered security as “security models, mechanisms, systems and software that have usability as a primary motivation or goal” [9]. The challenge is to give "end-users security controls they can understand and privacy they can control for the dynamic, pervasive computing environments of the future" [9]. This strongly reflects in the emerging field of identity management (IDM) as it is all about processing the user's personal data for identification. User-centric IDM has the central aim to give users control over their personal data and allow them to understand and manage how these data is being processed in different contexts [16].

The importance of privacy issues and an adequate consideration of principles for data protection and privacy are obvious. But the technical realization of privacy and security is a complex task and it is challenging to implement a system that is both - secure and usable. Security systems often suffer from an imbalance between usability and security. As an understated security level undermines the objectives of the system, this imbalance is in many cases at the expense of usability.

Moreover, Jøsang et al. also postulate “a very real difference” between the degree of security of a system in theory and its actual security. This underlines the potential trade-off between usability and theoretical security, as the intended protection of security systems strongly depends on the user’s understanding of the system [17]. The introduction of new security technologies such as e-IDMS brings further challenges to avoid this possible trade-off. Hence, usability “becomes a strategic issue in the establishment of user authentication methods” [18]. Generally speaking, a user-centric e-IDMS should provide privacy protection and security as well as usability. The incorporation of security usability is essential for the success of secure technologies.

B. Principles for Security Usability

The existence of principles for security usability indicates the demand for suitable approaches to avoid this possible trade-off. One ancient and important attempt that influenced security design was provided by the Dutch cryptographer Auguste Kerckhoff. Already in 1883, he described six principles for security systems: 1. The system must be substantially, if not mathematically, undecipherable; 2. The system must not require secrecy and can be stolen by the enemy without causing trouble; 3. It must be easy to communicate and remember the keys without requiring written notes, it must also be easy to change or modify the keys with different participants; 4. The system ought to be compatible with telegraph communication; 5. The system must be portable, and its use must not require more than one person; 6. Regarding the circumstances, in which such a system is applied, it must be easy to use and must neither require stress of mind nor the knowledge of a long series of rules.

These principles had high impact on today’s security and cryptography systems and despite of their age, some are still relevant. Jøsang et al. [17] underline the particular importance of principles 3 and 6 for today’s system design. They tied in with Kerckhoff's principles and developed principles for security usability. They distinguish between principles for security action and security conclusion. A security action is triggered, when the system demands the user to produce some information or set a security mechanism, (e.g., entering a password is a typical security action). Security conclusion means the users’ ability, to recognize the security state of the system (e.g., knowing that a connection via SSL uses encrypted data transmission). The principles are based on the conclusion, that the intended protection provided by a security system strongly depends on the user’s capability to understand, which security actions and conclusions the system requires and to react appropriately. “Security systems will only be able to provide the indented protection when people actually understand and are able to use them correctly” [17].

Another approach that deals with user-centric security is provided by Cameron’s seven laws of identity [6], which offers some important guidelines for user centricity in identity management systems. Similar to the principles of Jøsang et al., the rules for system design suggested by Cameron also focus on user understanding as a crucial factor for providing the intended level of security.

User understanding is definitely a crucial aspect for user centricity that often suffers from exaggerated security claims. Hence, some authors (cf. [19] [20] [21]) question the effectiveness of common security advice and principles in this respect. For instance, minimum requirements for password security (e.g., length, combination of signs and numbers, etc.) might be of vast importance in theory. However, as they are often not practicable and security risks are rather abstract to users, they are more of a burden for them. Most security advice are simply too complex for being useful to end-users and do not fit their demands on the system. As a result, security mechanisms foil themselves and systems are often insufficient regarding user experience.

IV. DETERMINANTS FOR A CITIZEN-CENTRIC IDM APPROACH IN E-GOVERNMENT

The previous remarks show a certain demand for a consideration of further aspects that go beyond technical design issues. As this paper deals with national IDM in the field of e-government we emphasize on identifying crucial factors for user centricity in this respect. In order to highlight how user centricity concerns the technical design of an e-IDMS, this Section starts with an overview of the basic models for identity management.

A. Evolution of Technical ID Models

There are four main types of ID models that can be distinguished: siloed, centralized, federated and user-centric systems. A siloed system is completely uncoupled from other systems with no formal connections with other IDMS. Hence, data processed within the IDMS is separated from other systems and cannot be easily linked across different domains. With respect to data protection and privacy this is a highly important aspect. However, due to this separation, a siloed system does not facilitate data sharing. Therefore it often does not fit the needs for an efficient data processing and sharing across multiple domains. Thus it allows no SSO either and users need multiple accounts when interacting with more than one system.

The *centralized approach* aims to ease this inconvenience. In centralized systems all of a person's data are stored in a repository managed by a central provider and independent from the applications using the data. This central repository is accessible to service providers, which can use it for their applications. Users are able to authenticate through one account. However, the potential threats to security and privacy are high in centralized systems as all personal data is being processed in one single unit and users are completely reliant on the central provider.

The *federated model* represents a sort of mixture between the siloed and the centralized approach. Here, a central identity provider (IdP) manages data relevant for identification of a person and providers of services and claims (SP) can use these data. The federation allows linking up previously unlinked identifiers and SPs base their applications on one single authentication mostly without creating or maintaining user accounts on their own. Users only authenticate via one single account, which can be used for multiple services. Hence, a federated system offers more user convenience and reduces privacy threats of the centralized model as the IdP normally does not hold all personal data. However, as the IdP knows, which identifiers belong to a specific person, he has the ability to abuse this knowledge and breach the user's privacy. Thus the functionality of a federated system strongly depends on the reliability of the IdP and the creation of a trustworthy infrastructure.

To diminish the users' dependence of a central IdP in a federated system, the *user-centric model* evolved. It has a certain focus on the person interacting with the system and offers her more control. There is no central IdP, users can choose between different SPs as well as IdPs. As identity providers dot not belong to a federation they are expected to act in the users' interest rather than in those of the SPs. Due to this freedom of choice, which parties to trust and which information to reveal in a particular transaction, a person is more independent and can gain advanced reliability in a user-centric system. However, this extent of control also brings greater demands on the users' skills to handle this [5] [8] [12].

B. (Preliminary) Parameters

The relevance of user centricity for the success of e-government is evident. Already in 2004, the mid-term review of the EU action plan eEurope 2005 attested a need for a "move to a demand-driven approach that emphasizes service delivery, end-user value for all and functionality" [22]. As citizens build a major subset of users in e-government, this paper focuses on citizen centricity in the context of electronic IDM and the term "user" mainly refers to the citizen. IDM in e-government is different from IDM in private sector. This difference demands for the consideration of other aspects. So as to realize a user/citizen-centric e-government approach, Blakemore and Undheim appeal for "a clearer focus on technologies that use citizen-relevant channels to deliver citizen/public value, rather than just to deliver efficiency gains and cost savings" [23]. Governments have to ensure equal access to public services for all citizens. This implies the multi-channel principle, i.e., to offer alternative channels to government services (online as well as offline). Online public services should be usable with familiar technologies in order to "maximize inclusion and utility, and to avoid unnecessary demands (skills, device purchase etc.) on citizens" [23].

In [1] we identified three major factors for citizen centricity in national IDM:

Equality of access: In e-government, IDM has to consider issues on a broader scope such as social inclusion, affordability, consistency, interoperability and the availability of public services for the whole population. Inclusiveness and providing non-exclusive access to public services via traditional as well as online channels to all citizens is a central requirement. Public services have to be accessible without e-ID as well and without any disadvantages in order to avoid a digital divide. The e-ID should reduce, not enlarge the distance between the citizens and public administration.

Privacy protection: The consideration of data protection and privacy aspects as a core issue is of vast importance for IDM. Governments have the substantial duty to protect the citizen's privacy and support them in controlling their personal data. Hence, a major requirement on a user-centric e-IDMS is its contribution to empowering users in managing their ID in a self-determined way. One crucial property "that must be satisfied in order to ensure privacy protection" is the unlinkability of personal data [24]. This means to avoid the use of unique identifiers, which are a threat to privacy because they can be used for „privacy-destroying linkage and aggregation of identity information across data contexts" [25]. Thus, different identifiers for every sector should be used, e.g., in the form of local pseudonyms [25]. Data processing in the e-IDMS has to be transparent to users so that they are able to comprehend how their personal data is being processed within the system.

Citizen convenience: Improving convenience for users is a central issue for IDM, as it determines how the system responds to the citizens' demands. This affects e-government in particular as public services should be usable for every citizen. The e-ID should support SSO and ease the users' need to handle multiple accounts and the corresponding procedures. At the same time the e-ID should provide citizens' a suitable and convenient way to deploy their e-ID in different contexts without the need for handling multiple login data and procedures.

An additional factor in line with these parameters is *trust.* It is strongly interrelated with the other factors and particularly connected to the possible trade-off between security and usability. One might say trust is in between the poles of this possible trade-off: Security and trust are interdependent and determine each other in some respects. Lacking usability foils security and as a consequence also lowers trust in the system. When security mechanisms empty into high complexity, users are then not able to understand and consider them appropriately. One important aim of national IDM is to increase the amount of trust in e-government. “Concepts of trust and identity have become intimately bound, and go beyond a purely technical focus” [26]. Government organizations require citizens to trust in them “in order to be legitimate and efficient” [27]. Applying a reliable environment for public service usage is an essential precondition for citizens’ trust in e-government. The e-IDMS should establish a solid fundament for trustworthy interactions between citizens and government with the assurance, that his personal data is treated correctly and not against his privacy [27].

V. Development of the Austrian e-IDMS and Specific Design Characteristics

First initiatives for a national e-IDMS already began in the early 90ies with plans to set up a smart card system in the field of social and health insurance administration. The European Directive (1999/93/EC of December 13 1999 on a Community Framework) for electronic signatures triggered further impulses at a European level. Austria was (among other EU-member states) directly involved in designing the signature and hence one of the first countries in Europe to implement a national e-IDMS for e-government services. In October 2000, the idea of a smart card for unique identification of citizens in a certain role – the so-called “Citizen Card” (in German called “Buergerkarte”) was born and announced as an integral part of Austria’s national conversion of the eEurope initiative “information society for all”. Shortly afterwards the government approved a resolution for the implementation of a smart card based system to support e-government services [28]. First prototypes of Citizen Cards were released during a pilot scheme and available from 2002 until 2005.

As the system architecture for the Citizen Card (CC) follows a technology-neutral approach the concept is not bound to one specific card. Although plans during the development process aimed to use the electronic health insurance card (today known as “e-Card”) as primary device for the CC concept. Together with the ATM card, the e-Card became one of the major carrier devices to carry the CC-function.

A. Major system characteristics

The Citizen Card as centerpiece of the Austrian e-IDMS has some specific characteristics. First of all, it strives for technology-neutrality and multiple tokens as it is not a physical card but a virtual concept that can be implemented on various different hardware components (e.g., smart cards, cell phones, USB devices) [29] . Due to their broad range of use, smart cards are currently the preferred carrier devices with e-Cards and ATM cards as main tokens. These cards are wide-spread among the Austrian population. Every citizen (8.3 million) has an e-Card and about 80% of the Austrians hold an ATM card. These cards have the “sleeping” CC-function integrated, which means that they are prepared for the e-ID but the function needs initial activation. Ministerial IDs, staff IDs of the Chamber of Commerce and student IDs are some of the further possible carrier devices. The Citizen Card fulfills two basic functions in online transactions with public administration: it allows to verify the card holder’s identity and to authenticate his/her request by providing an electronic signature, which is stored on the card. A peculiarity of the e-IDMS is its ID model and the technical privacy concept: the system is based on a complex techno-organizational infrastructure with an ID model that is grounded on unique identifiers in the Central Register of Residents (CRR), whereas sector specific identifiers (ssPINs) are derived from. The amount of data stored on the card depends on the specific carrier device. But every Citizen Card contains at least the card holder’s full name, date of birth, the source-PIN as unique identifier (for details see next Section) and the cryptographic public keys needed for the e-signature and content-encryption. The private key is stored in a separate hardware unite on the cards’ chip. For protection of these data, up to three different PIN-codes that are only known by the card holder are applied. The first one is for general access protection of the device, the second one for using the e-signature and the optional third for the additional feature of an integrated data box for storing electronic documents such as a birth certificate [29] [30].

B. Techno-organizational infrastructure and ID model

The Austrian e-IDMS is based on a complex techno-organizational infrastructure. This set-up can be explained regarding the CC’s two main functions – identification and e-signature. For the creation and provision of the e-signature, a Public key infrastructure (PKI) was established. The PKI consists of one or more Certificate Authorities (CAs) that issues all services relevant for the e-signature and Registration Authorities (RAs), where card holders can apply for an e-signature. (Currently, the institution a.trust is the only CA in Austria that applies qualified certificates required for the e-ID). This CA coordinates several RAs (i.e., banks, post offices, etc.), which usually provide the full activation of a Citizen Card including the integration of the ID model [29] [30].

The core infrastructure component for the ID model is the Central Register of Residents (CRR). This register is a national database, which contains data of all Austrian residents. The primary key for every data-record is the CRR-number, a 12-digit number, which acts as unique identifier for a specific person. The CC’s whole ID model is based upon the CRR-No. but not directly used to respect privacy protection. Hence only a strong encryption of the CRR-No. – the so-called source-PIN – is stored in the card to identify the card holder and the law prohibits storing it outside the card.

The source-PIN is created during card activation and used for generating sector-specific PINs (ssPIN). An ssPIN is based on an irreversible cryptographic function, which prevents to recreate its original elements (i.e., the source-PIN). Currently, there exist ssPINs for 26 sectors (e.g., tax, education, health, etc.). An ssPIN is used for unique identification of a person within the specific belong sector. Storage of an ssPIN is regulated by the law and only allowed within the sector it belongs to or is allowed to use it [30]. The Figure below gives an overview of the interrelations between the major infrastructure components.

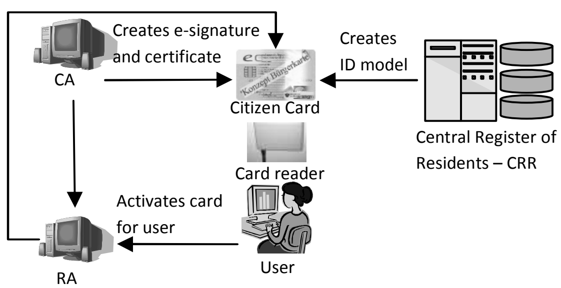


Figure 1: Techno-organizational infrastructure

C. Requirements and user interaction with the e-IDMS

Using the CC in online services requires the initial activation of the function. RAs carry out the corresponding procedures for the card holder. Until recently, this was only possible by visiting an office. Now the whole activation process can be carried out online as well with the precondition that the e-Card is the carrier device. For the handling of the CC, a PC with internet connection, a card reader and special software – the so-called Citizen Card environment (CCE) – are required. The CCE is available in different variants, including productions completely free of charge. In 2009, an online-variant of the software has been introduced. The activation for ATM cards costs 12 € once, and the certificate for the e-signature is 15.60 € per year.

A typical user session with CC usually proceeds as follows: most public online services are available via SSO on the Austrian e-government portal help.gv.at. After choosing a service, the user is prompted to authenticate by putting his card into the card reader and entering a PIN-code. This grants the service access to the user’s ID data on the card in order to generate a confirmation for accessing the service (typically, it looks like this: “I, John Doe, born on January 1st 1973, confirm that I am using this service. Date, time: January 12, 2010, 9:32:12”). This confirmation has to be signed by the user by entering his signature-PIN. Then, depending on the current service, some forms have to be filled out with personal data (e.g., income data for tax declaration). When submitting, the user is prompted again to sign another confirmation in order to affirm his service request and the correctness of his data. During submission, the service requests creation of the ssPIN in the back office, by reading the source-PIN out of the card and combining it with the unique number of the current sector the service belongs to (e.g., tax). After service completion the data is being further processed in the back office applications of the appropriate authority. It depends on the administrative procedure, whether data processing is completely automated or includes further treatment by the administration office [29] [30].

VI. THE BIASED PICTURE OF A CITIZEN-CENTRIC VIEW ON THE AUSTRIAN E-IDMS

This Section analyzes the realization of relevant determinants for citizen centricity (as described in Section IV) in the Austrian system and then strives for explanations of the biased picture that the analysis and the following delineations draw.

A. Mapping of the e-IDMS against typical ID models

A tentative attribution of the Austrian system to the four basic ID models as described in Section IV.A is revealed by the Table below:

TABLE I. MAPPING OF THE E-IDMS

	Siloed	Centra -lized	Fede-rated	User-centric
Method of authentication			X	
Location of Identity Information		X	X	(X)
Method of linking accounts/learning if they belong to the same person			X	(X)
Trust Characteristics (who is dependent on whom, for what)		(X)	X	
Convenience			X	(X)
Vulnerabilities	(X)	X	X	

As the basic models represent rather simplified “ideal-types”, this mapping cannot be stringent, but still it gives an initial clue about the system with regard to citizen centricity.

The user authenticates with the Citizen Card to each service via SSO, whereas the CRR is the central identity provider and supplies the ID data to service providers. Hence, the method of authentication correlates with the federated model. Regarding location of identity information, the e-IDMS is mainly a mixture of centralized, federated and user-centric model: all relevant ID information is centrally stored in the CRR that service providers can integrate in their separate accounts but identity verification usually requires the CC. Linking of data across different domains is prevented by the ssPINs and the corresponding legal regulations, i.e., it is only allowed to link data within the same sector or by offices permitted to process the data. But the identity provider knows in which services a user deploys his e-ID and the CRR contains more personal data than generally needed for every service. Thus, users have to trust that the federal identity provider and the service providers use their data properly and with respect to privacy. Due to its powerful role, the user is somewhat constrained by the federal identity provider. As service utilization with e-ID requires the CC as a separate device and PIN-codes, users gain more control. However at the same time they are also confronted with increased requirements to handle the e-ID. As SPs have to request the federal IdP for the creation of ssPINs, they are not completely liberated from the burden of credential management. Altogether, the Austrian e-IDMS mainly follows a federated approach, whereas some of its features strive for a user-centric system design.

B. Provision for citizen centricity

The e-IDMS incorporates a citizen-centric approach and the consideration of relevant aspects reflects in several ways. Regarding *equality of access*, different approaches have been employed to avoid social exclusion and exclusiveness of the e-IDMS; some of them especially during the rollout phase also in order to broaden penetration and stimulate usage of the system: to reduce financial burden, online transactions with the e-ID were free of charge until the end of 2006. Since 2008, there are no costs when using the e-Card as CC. The technology-neutral concept allows using several different devices as carrier for the e-ID. Due to the possibility to integrate the e-ID into different systems, it provides openness and interoperability at least to some extent. The e-ID is not compulsory and citizens are free to decide whether to use it or not. Austria provides a broad scope of different e-government services and of course, services are still available in traditional offline forms. Online services do not per se require the e-ID and can be used with common authentication methods (i.e., username/password) as well. Just a few services require the e-ID and only in cases, where the transaction should be processed completely online without any media friction. In this respect, the e-ID could also be noticed as enabler of an additional access-channel. Due to the availability of multiple tokens, citizens also have some choice, which carrier device to use as Citizen Card. As there are no costs for using the e-Card, neither for activation nor for usage, most people are expected to prefer this device.

For the consideration of *privacy*, the e-IDMS is based on a sophisticated ID model, which strives for a balance of security and data protection. As persistent static identifiers allow data linking across different domains, they enable potential privacy threats, i.e., identity fraud or infringement of personal information [4]. Hence, the Austrian e-IDMS is based on a complex ID model, which avoids the direct processing of a unique identifier (as described in Section V). The use of the ssPINs aims to prevent illegal linkage of personal data. These identifiers are different for a defined number of domains (currently 26), and legal regulations limit the use of an ssPIN to the domain it origins from or is allowed to use it. Moreover, it is also prohibited to persistently store the source-PIN (as basis-number for an ssPIN) outside the Citizen Card. This technical sector separation corresponds to the deployment of different pseudonyms. As the e-IDMS applies an electronic token in form of a hardware device, users receive at least some control over their personal data. The combination of knowledge (the PIN-code) and possession (the card) improves security of the authentication procedure compared to usual concepts, which are based on username/password.

The system contributes to enhance *citizen convenience* as it provides a comprehensive approach to harmonize authentication procedures. Most Austrian e-government services are available at the e-government portal help.gv.at and citizens can use their CC to authenticate at this single entry-point via SSO. With the CC as one device to authenticate in different services, identity management is alleviated as citizens do not have to handle several user accounts and credentials. The openness of the concept to different carrier devices gives users the possibility to choose their preferred token for the CC-function. The possibility of activating the e-Card completely online offers a convenient way to enable it as carrier medium. Beside the two main carrier cards (e-Cards, ATM cards) there was also a CC available on a cell phone without needing a smart card or card reader. A legal provision allowed this so-called "Citizen Card light", which had less security requirements. As the legal regulation was only temporarily, the "Citizen Card light" was only available until the end of 2007. In November 2009, an improved version of the CC on a cell phone has been announced and is available since 2010 [31]. An additional online version of the CCE is available since 2009, which is completely browser-based and thus reduces efforts as users do not have to install additional software components.

The techno-organizational infrastructure of the e-IDMS contributes to create a circle of *trust*. The involved parties (CA, RA, IdP, SPs) have to fulfill certain requirements, which are legally defined (e.g., in legal regulations for e-signature, privacy, administrative procedures, etc.). The Data protection commission serves as a custodian over the lawful appliance of the e-IDMS. The Ministry of the Interior administrates the CRR and acts as central identity provider on behalf of the DPC. Service providers have to register their applications and to request for deploying their services with the CC. Due to the privacy aware system implementation and the increased amount of control, the e-IDMS seems to be grounded on a reliable fundament that is capable of enhancing citizens' trust in e-government.

C. Current usage and acceptance of the system

With the two main carrier devices – e-Card and ATM cards – the penetration of potential CC is high as these cards are wide-spread in Austria and already prepared for the CC-function. There is also a number of services available that can be used with the CC at all three administrative levels (federal, provincial, municipal) at the Austrian e-government portal. However, there is no significant increase in card activation and usage although the Citizen Card is obtainable for several years already. The optimistic goals for the number of card activations had to be adjusted downwards several times. E.g., the intended number of 200,000 active CCs by the end of 2005 was not achieved. In 2006, only about 60,000 activated cards were in use and a substantial part of these are bulk activations by public organizations [1]

[32]. According to recent estimates of the Federal Chancellery, about 120,000 were circulating by May 2009 [33].

A look at the usage levels of three exemplarily online services reveals some peculiarities [1]:

TABLE II. USAGE OF SERVICES IN 2007

Transactions	Total	Total - online	Citizen Card
Tax declarations	Approx. 4,000.000	1,846.922 (46%)	12.801 (0.7%)
Student grants*)	66.933	53 (0.1%)	53 (0.1%)
Retirement pay account **)	35.974	10.485 (29.1%)	10.485 (29.1%)

*) Data refer to academic year 2007/08. **) Data refer to 2008.

There are some remarkable differences in usage of these services. The number of transactions for tax declaration is of particular interest as it is considerably higher compared to the other services. It is the most successful e-government application in Austria; the amount of citizens transmitting their tax declaration online is close to 50%. However, the vast majority prefers common authentication based on username and password. Less than 1% uses the CC for this service. As online processing of the two other services requires authentication with CC, the number of online transactions equals the number of transactions with Citizen Card. The comparatively higher amount of citizens using the online service for retirement pay account queries is explainable by the significantly higher number of potential users (students only represent a small share of the population) as well as strong advertising and PR actions taken for e-health and social services during the roll out of the e-Card, which received increased public attention.

D. *Explaining the current situation*

The usage level is only progressing very slowly and a remarkable increase as expected by the main stakeholders has not occurred yet. Despite of the important considerations of citizen centricity in the Austrian system, the overall situation draws a rather biased picture.

In order to identify explanations for this situation it is expedient to take up a more general view on the e-IDMS as a security system, as this allows gaining a better understanding of relevant determinants. From this point of view, the e-ID represents a certain security mechanism for citizens when interacting with public administration. Users are mostly considered to be the soft spot of a security system that often neglect the proper use of security mechanisms. This negligence is often stated as irrational or ascribed to the users' lack of understanding the security mechanism. However, a closer look shows that users act "entirely rational" when rejecting security advice, as [21] argues: "A main part of the problem with security advice is that we hugely exaggerate benefits". Additionally, the cost of user effort is often ignored [21]. Security systems and the corresponding requirements overwhelm users and offer them "a poor cost-benefit tradeoff". Therefore, security mechanisms are often rejected by users as the high requirements made of users do not match the predicted benefits. Users are confronted with a real effort to handle a security system while at the same time this effort should prevent from threats that are rather theoretical [20] [21]. Applications with an exclusive focus on security mostly offer "a small perceived advantage in exchange for dealing with an extraordinary complex interface" [19].

These remarks can also be transferred to the situation of the Austrian system. The high complexity of the e-IDMS plays a certain role for the user experience and thus is a major vulnerability of the system, which has been a central point of criticism and entails further controversial aspects. Although the sophisticated ID model was designed to prevent data linkage and protect the citizens' privacy, the effectiveness of this solution is questioned. As online service process many personal data, illegal data linkage is still feasible over these data, despite of the deployment of ssPINs. The complex coherences of the system cause a lack of transparency, which does not allow users to comprehend how their e-ID and the related data are being processed within the system. This also limits the users' amount of control over their personal data. Essential requirements for preventing the e-ID to become an instrument of surveillance are effective controls of the maintenance of fundamental privacy principles (e.g., commensurability, data minimization, purpose limitation of data processing, etc.). Due to the high overall complexity and opacity of the system, this controllability of a proper data processing in account with privacy is rather hard to ensure. Lacking transparency and high complexity can also be expected to lower the citizens' level of trust in the e-IDMS. Overall, the e-IDMS and especially the CC are perceived as too complex with several flaws regarding citizen centricity. Benefits and convenience are rather low compared to the high requirements made of users [32] [34].

These propositions address several serious aspects, which indicate the suspense between usability and security in the Austrian system in several contexts. There is some certain evidence for this assessment. Two studies revealed some interesting indications for the situation: In the "eUser" study of 2005, 27% of the Austrian Internet users described the need for a special end-user device for identification (i.e., the CC) as a burden for using online public services. Costs were estimated as too high compared to the expected benefits of the CC [35]. According to another study from 2006, 33% of the respondents mentioned to have no intention to obtain a CC at all. The reasons stated for this correspond to the propositions above: There is no or not enough need for the CC (46%), lack of information about usage (37%), the CC is not trustworthy enough (22%). Furthermore, 38% of the respondents that stated to be card holders mentioned to never have used their CC [36].

When considering the high requirements for usage (card activation, card reader, installation of special software), it is not very unlikely that handling of the e-ID is perceived as burden. Several problems and obstacles in practical use also appear from entries in the online support-forum for CC users.[1] Practical tests conducted for this research confirmed the non-trivial and partly complicated handling of the e-ID. Indeed, the number of security actions and conclusions (cf. [16]) demanded from the user often seems to be over exaggerated and beyond a standard users understanding of a common system.

Beside the problem of high complexity, from a user's perspective, the system does not seem to offer enough benefits and incentives. The marginal rate of contacts for a citizen with public administration (only approx. 1.7 contacts per year) and the existence of common and fairly effective authentication methods are important aspects in this context [1]. This, combined with the exaggerated efforts of using the e-ID may considerably account for a weak benefit/cost ratio (whereas costs do not primarily address financial expense, but a disproportional effort). Thus, the e-IDMS provides only a low user value.

It has to be noted that major stakeholders are aware of this situation and since the mentioned studies have been conducted, several measures were set to improve citizen centricity and increase diffusion. The measures mainly address the reduction of costs and usability problems: e.g., no charges for the e-Card, promotion of cost-reduced notebooks with integrated card reader and pre-installed software, an additional online version of the CCE and the re-launched option of a cell phone based CC. To increase penetration and usage, several promotion campaigns mainly target teenagers and students. However, at present it is uncertain whether these actions are adequate to cope with the current situation and increase the level of usage.

VII. IMPROVING USER CENTRICITY BY (RE-)FOCUSING ON THE USER VALUE

The previous Section has shown that, although the Austrian system seems to consider several citizen-centric aspects the actual situation is not satisfying regarding usage and acceptance. This leads to the TAM as its aim is to identify relevant determinants for acceptance or rejection of a technology. A classification of the current situation to the two factors of the TAM, usefulness and ease of use, confirms the biased picture depicted and offers further explanations. The benefits a user expects from the system are addressed by the factor perceived usefulness. The ease of use addresses the costs and efforts that system usage entails. Now it has been pointed out that these efforts are perceived as rather high because citizens are confronted with additional requirements (i.e., card activation, card reader, special software) and the overall complexity of the system is perceived as too high.

At a first glance, the reduction of complexity might come into mind as necessary approach for easing the situation and improving citizen centricity. Reducing complexity certainly is important to lower current burdens to usage. But does this also stimulate usage? At least in the Austrian case this effect did not occur: stakeholders took several measures in this regard to alleviate handling of the e-ID. However, it is currently not foreseeable whether these actions will be effective. Moreover, the scope of action for reducing complexity might be limited with respect to the intended security level. Plus, a deviation from the sophisticated ID model of the Austrian e-IDMS cannot be expected to be performed easily without enormous efforts and problems regarding privacy protection.

In this regard, a rather interesting aspect is pointed out by Gutmann and Grigg: users do accept "a little more complexity (…) for a fair offering in value" [18]. Davis [13] argued similar, whereby "(…) users are often willing to cope with some difficulty of use in a system that provides critically needed functionality". This implies a stronger focus on finding a balance between acceptable complexity and user value. It might seem obvious that system usage is strongly interdependent with the benefits users can expect. But as already underlined in the previous Section, this determining issue seems to be neglected especially in terms of security and identity management systems. Hence, "cost and benefits have to be those the users care about, not those we think the user ought to care about" [20]. Costs are not just meant in a monetary sense here but subsume all the efforts that users are confronted with. When the transaction costs incurred by switching from familiar forms of identification to the e-ID are high and the expected benefits due to this switching are low then usage is expected to be low either.

In accord with the TAM it thus strongly depends on the perceived usefulness of the e-IDMS, whether users are willing to accept a certain degree of complexity. The difficulty of usage can surely contribute a lot to "discourage adoption of an otherwise useful system", but "no amount of ease of use can compensate for a system that does not perform a useful function" [13].

The analysis has already shown that the system does not seem to offer enough benefits and incentives. Whilst the scope of available services is relatively broad, at the same time, the average frequency of citizen contacts per year is relatively low. Hence the incentives to access these services via CC are marginal and the usefulness is perceived as too low either.

When considering the recent measures of major stakeholders, it is salient that the ease of use, respectively the usability of the e-IDMS seems to receive more attention than usefulness. Hence there is a certain demand for increasing benefits and creating a "real" user value, which implies, that service provision plays a crucial role for user centricity in the e-IDMS. From a user's point of view, current services with e-ID do not considerably differ from common e-government services except of the authentication method. An important step forward might be finding out, which additional benefit of the e-ID citizens would really appreciate. For instance services, that offer new possibilities for interaction with public administration and that legitimate the sophisticated concept behind the system. At its current state, the e-IDMS seems to be less suitable as an instrument for standard-users than for users with special demands. For instance, the number of citizens with a frequent use for the e-signature and document encryption yet seems rather marginal. This might be different in businesses with a certain demand for this application and the security level provided by the e-ID.

[1] http://tinyurl.com/c9kuvn

Major stakeholders also rated the business sector as crucial for further diffusion.

A certain additional value for citizens could be to enhance transparency of government actions: e.g., to grant users access to administrative documents that pertain to themselves and to provide information about current administrative proceedings in terms of freedom of information, of course with respect to privacy and data protection issues. Here, trust as an important aspect comes in again. Freedom of information laws appears “to have contributed to citizens showing higher levels of comfort about how their information will be handled" [4]. Or in other words: when citizens are able to comprehend how their personal data is being processed in public administration, this contributes to increase trustworthiness in government, which represents an important incentive. From a privacy perspective, transparency is essential as surveillance can only be effectively controlled and prevented when information about purpose of e-ID usage and processing are definitely regulated and accessible for citizens [34]. Applications for e-ID in terms of freedom of information would also be conducive to improve effectiveness of privacy as it contributes to improve an individuals’ control over his e-ID respectively his personal data.

VIII. CONCLUSION

The analysis of the Austrian e-IDMS regarding its incorporation of citizen centricity reveals a biased picture: although the system includes important citizen-centric factors and several measures were set to reduce complexity and alleviate handling of the e-ID, the level of usage and acceptance of the system does not meet the expectations of major stakeholders. This ambivalent result highlights that the intentions behind the e-IDMS regarding end-users do not seem to match the users’ perceptions of the system. The implementation of the Austrian system was dominated by strong focus on security. This entailed a high overall complexity, which is a particular burden for acceptance and usage. However, measures to reduce this complexity have not lead to the intended effects yet. In this respect, the e-IDMS indeed reflects the depicted suspense between security and usability. The crucial challenge for security systems in general and e-IDMS in particular is to find suitable approaches for avoiding this suspense. First and foremost this implies a stronger focus on providing a “real” user value. This seemingly rather obvious finding addresses the necessity for a paradigm shift in system design to compensate the mismatch between design philosophy behind the system and the usability needs regarding security usability from a user’s view.

An important step towards finding suitable approaches for easing this situation is to emphasize on user perception as determinant of vast importance for user centricity. The deployment of the TAM allowed to conclude that this necessary focus is currently rather neglected in the e-IDMS. The measures taken mainly address the ease of usage (i.e., increase usability) whilst the usefulness of the system (i.e., the expected benefits) is left behind. Hence, there is a certain demand for further efforts to improve usefulness, which mainly concerns service provision and the creation of additional user value. Whilst the e-ID in its current state inter alia suffers from the end-users irregular demand, businesses might have a more frequent need.

The Austrian case-study provided a useful example about the importance of considering further aspects in system design that go beyond technical issues in order to gain an expedient level of user centricity. These aspects refer to complex interrelations among multiple scopes especially in terms of e-government. The major challenge in this regard is to balance multiple goals, i.e., to provide a certain level of security, protect the citizens’ privacy and offer both usable and useful features from a citizen’s perspective. Necessary adaptations do not just address the technical design of the system but might also include further actions to take, i.e., reconfigurations of policy frameworks and legal regulations for e-ID usage. Furthermore, governments neither are nor should they be in a position to simply introduce additional services as this is a matter of checks and balances. Hence, a focus on improving transparency for citizens could contribute to experience user value. An additional benefit citizens could appreciate might be to facilitate access to government information and administrative proceedings that concern them. This institutionalization of freedom of information would also contribute to improve trust in (e-) government. As the e-IDMS represents an innovation, a rather tentative increase in acceptance and usage does not seem surprising. However, when this technology should establish itself in a mid-term perspective, this requires by all means the composition of further measures in order to strive for a more balanced provision of citizen centricity with respect to its multiple determinants.

REFERENCES

[1] G. Aichholzer and S. Strauß, "The Citizen’s Role in National Electronic Identity Management. A Case-study on Austria", Proc. of the Second International Conference on Advances in Human-Oriented and Personalized Mechanisms, Technologies, and Services (CENTRIC 09), IEEE Computer Society, 2009, pp. 45-50, doi: doi.ieeecomputersociety.org/10.1109/CENTRIC.2009.13

[2] M. Lips, J. Taylor, and J. Organ, "Identity management as public innovation. Looking beyond ID cards and authentication systems", in ICT and public innovation: assessing the modernisation of public administration, V. Bekkers, H. van Duivenboden, and M. Thaens, Eds., Amsterdam: IOS Press, 2006, pp. 204-216.

[3] M. Lips, J. Taylor, and J. Organ, "Electronic government towards new forms of authentication, citizenship and governance", Paper to Oxford Internet Institute conference: Safety and Security in a Networked World, October 8-10, Oxford University, 2005.

[4] R. McKenzie, M. Crompton, and C. Wallis, "Use Cases for Identity Management in E-Government", in IEEE Security and Privacy vol. 6 (2), March-April 2008, pp. 51-57.

[5] A. Josang, M. AlZomai, and S. Suriadi, "Usability and Privacy in Identity Management Architectures", Proc. of the fifth Australasian symposium on ACSW frontiers - Volume 68, Australian Computer Society, 2007, pp. 143-152.

[6] K. Cameron, "The laws of identity", 2005, http://tinyurl.com/2226ry.

[7] J. S. Pettersson, S. Fischer-Hübner, N. Danielsson, J. Nilsson, M. Bergmann, S. Clauss, T. Kriegelstein, and H. Krasemann, "Making PRIME usable", Proc. of the 2005 Symposium on Usable Privacy and Security (SOUPS '05), vol. 93, New York: ACM Press, 2005, pp. 53-64. doi: doi.acm.org/10.1145/1073001.1073007

[8] A. Bhargav-Spantzel, J. Camenisch, T. Gross, and D. Sommer, "User Centricity: A taxonomy and open issues", Journal of Computer Security vol. 15 (5), October 2007, pp. 493-527.

[9] M. E. Zurko, "User-Centered Security: Stepping Up to the Grand Challenge", Proc. of the 1st Annual Computer Security Applications Conference (ACSAC 05), IEEE Computer Society, 2005, pp. 187-202, doi:doi.ieeecomputersociety.org/10.1109/CSAC.2005.60

[10] T. M. Eap, M. Hatala, and D. Gasevic, "Enabling User Control with Personal Identity Management", Proc. of the IEEE International Conference on Services Computing (SCC 07), IEEE Press, 2007, pp. 60-67, doi: doi.ieeecomputersociety.org/10.1109/SCC.2007.56

[11] Cap Gemini Ernst and Young, "Online Availability of Public Services: How is Europe Progressing? Web based Survey on Electronic Public Services. Report of the Sixth Measurement", 2006, http://ec.europa.eu/information_society/eeurope/i2010/docs/benchmarking/online_availability_2006.pdf

[12] M. Donohue and A. Carblanc, "The role of digital identity management in the internet economy: a primer for policymakers", No. DSTI/ICCP/REG(2008)10/REV1: Organisation for Economic Co-operation and Development (OECD), 2009.

[13] F. D. Davis, "Perceived usefulness, perceived ease of use, and user acceptance of information technology", MIS Quarterly, 13, September 1989, pp. 319-339.

[14] L. Leong, "Theoretical models in IS research and the technology acceptance model (TAM)", in Technologies & Methodologies For Evaluating information Technology in Business, C. K. Davis, ed., Hershey, PA: IGI Publishing, 2003, pp. 1-31.

[15] Y. Malhotra and D. F. Galletta, "Extending the Technology Acceptance Model to Account for Social Influence: Theoretical Bases and Empirical Validation", Proc. of the Thirty-Second Annual Hawaii International Conference on System Sciences (HICSS 99), Washington, DC: IEEE Computer Society, 1999, pp. 6-14.

[16] M. Hansen, H. Krasemann, M. Rost, and R. Genghini, 2003, "Datenschutzaspekte von Identitätsmanagementsystemen. Recht und Praxis in Europa", Datenschutz und Datensicherheit 27 (9), 2003, pp. 551-555, https://www.datenschutzzentrum.de/projekte/idmanage/DUD-27-9.pdf

[17] A. Jøsang, B. AlFayyadh, T. Grandison, M. AlZomai, and J. McNamara, "Security Usability Principles for Vulnerability Analysis and Risk Assessment", Proc. of the 23rd Annual Computer Security Applications Conference, 2007, pp. 269-278, doi: 10.1109/ACSAC.2007.14

[18] C. Braz and J.-M. Robert, "Security and Usablity: The Case of the User Authentication Methods", Proc. of the 18th International Conference of the Association Francophone d'Interaction Homme-Machine (IHM 06); Vol. 133, 2006, pp. 199-203, http://doi.acm.org/10.1145/1132736.1132768

[19] P. Gutmann and I. Grigg, "Security Usability", IEEE Security and Privacy 3 (4), 2005, pp. 56-58, http://www2.computer.org/portal/web/csdl/abs/mags/sp/2005/04/j4056abs.htm

[20] B. Schneier, "Balancing Security and Usability in Authentication", February 2009, http://www.schneier.com/blog/archives/2009/02/balancing_secur.html

[21] C. Herley, "So Long And No Thanks for the Externalities: The Rational Rejection of Security Advice by Users", Proc. of the 2009 New Security Paradigms Workshop (NSPW 09), September 2009, pp. 133-144, http://research.microsoft.com/en-us/um/people/cormac/papers/2009/solongandnothanks.pdf

[22] EU-Commission, "eEurope 2005 Mid-term Review", 18.02.2004, Bruessels: Commission of the European Communities, 2004, http://ec.europa.eu/information_society/eeurope/2005/doc/all_about/acte_en_version_finale.pdf

[23] M. Blakemore and A. T. Undheim, "A Handbook for Citizen-centric eGovernment", 2007, http://www.ccegov.eu/downloads/Handbook_Final_031207.pdf

[24] FIDIS, "D13.6: Privacy modelling and identity", Deliverable-Report., Future of Identity in the Information Society, 2007, http://www.fidis.net/fileadmin/fidis/deliverables/fidis-wp13-del13.6_Privacy_modelling_and_identity.pdf

[25] M. Rundle, B. Blakley, J. Broberg, A. Nadalin, D. Olds, M. Ruddy, M. T. M. Guimarares, and P. Trevithick, At a crossroads: "Personhood" and digital identity in the information society, No. JT03241547, 29.02.2008: Organisation for Economic Co-operation and Development (OECD), 2008, http://www.oecd.org/dataoecd/31/6/40204773.doc.

[26] F. Wilson, "Think Paper 11: Trust and Identity in Interactive Services: Technical and Societal Challenges", 2007, http://www.ccegov.eu/Downloads/Paper%2011%20-%20Trust%20and%20identity%20in%20interactive.pdf

[27] P. Chadwick, "Managing identity, privacy and the public trust, Institute of Public Administration Austrialia series: The art of the long view. The role of Government: Taking responsibility or sharing it?", 2006, http://www.privacy.vic.gov.au/privacy/web.nsf/download/D9010341D53F4684CA2571790011A65E/$FILE/Managing%20iden

tity,%20privacy%20and%20the%20public%20trust%20%28IPAA%29%2016%20May%202006.pdf

[28] A. Hollosi, H. Leitold, and R. Posch, "Die Bürgerkarte: Basis und Infrastruktur für sicheres e-Government", in Tagungsband der Arbeitskonferenz Enterprise Security - Unternehmensweite IT-Sicherheit, P. Horster, ed., Paderborn, Germany: IT-Verlag, March 2002, pp. 1-12.

[29] G. Karlinger, D. Konrad and R. Posch, "Weissuch Bürgerkarte", Version May 2002, http://www.buergerkarte.at/regain/file/D:/webs/at.buergerkarte.downloads/WeissbuchBuergerkarte.20020515.pdf?index=main

[30] Austrian Federal Chancellery,"Administration on the Net - The ABC guide of eGovernment in Austria", No. New edition 7/2008, Vienna: Austrian Federal Chancellery, 2008, http://www.digitales.oesterreich.gv.at/DocView.axd?CobId=19394

[31] Heise Online, "Österreich führt Handy-Signatur im E-Government ein", press release of June 07 2010, http://www.heise.de/newsticker/meldung/Oesterreich-fuehrt-Handy-Signatur-im-E-Government-ein-1016339.html

[32] ARGE-Daten - Österreichische Gesellschaft für Datenschutz (Austrian Society for Pricacy), "Was wurde aus der Bürgerkarte", press release of October 5 2006, http://www2.argedaten.at/php/cms_monitor.php?q=PUB-TEXT-ARGEDATEN&s=50759zjj

[33] ORF Futurezone, "E-Voting: Blackbox statt Wahlkabine", press release of May 17 2009, http://futurezone.orf.at/stories/1603378/

[34] S. Priglinger, "Auswirkungen der EU-DL Richtlinie auf die E-Gov-Welt", in Datenschutzrecht und E-Government, D. Jahnel, ed., Graz, 2008, pp. 267-283.

[35] eUser study, "Public online services and user orientation - country brief Austria", 2005, http://tinyurl.com/cj3hus

[36] Fessel GfK - Institut für Marktforschung Ges.m.b.H. (Institute for market research),"Online Studie 6 Monitoring E-Government", 2006, http://www.digitales.oesterreich.gv.at/DocView.axd?CobId=21828

[37] C. N. M. Pounder, "Nine principles for assessing whether privacy is protected in a surveillance society", Identity in the information society (IDIS) vol. 1 (1), 2008, pp. 1-22.

Al URLs last visited on 30.06.2010

Critical Friend Model: A Vision Towards Inter-cooperative Grid Communities

Ye Huang*, Nik Bessis†, Amos Brocco*, Pierre Kuonen‡ and Beat Hirsbrunner*
*Department of Informatics, University of Fribourg, Switzerland
Email: {ye.huang, amos.brocco, beat.hirsbrunner}@unifr.ch
†Department of Computer Science and Technology, University of Bedfordshire, UK
Email: nik.bessis@beds.ac.uk
‡Department of Information and Communication Technologies,
University of Applied Sciences Western Switzerland
Email: pierre.kuonen@hefr.ch

***Abstract*—Much work is under way within the distributed computing community in order to assign a job to an appropriate resource discovered from a fully decentralized and heterogeneous infrastructure with reasonable cost, such as optimized job responsible time, executing price, etc. However, local resources of individual virtual organizations (VOs) are managed under independent policies and constraints, therefore existing solutions are normally designed for specific scenarios and lack of commonality. In addition, boundaries of different VOs raise extra difficulties on job sharing and collaboration amongst distributed nodes. On the other hand, the obtained knowledge from multi-node cooperations is normally discarded, although it in future may lead to intelligent scheduling decision by means of previous collaboration records and experience. Especially, the trust built up according to historical collaboration between nodes from different VOs may overweigh and cross the boundaries of VOs themselves. In this work, the Critical Friend Model (CFM) grid scheduling solution is proposed to bridge the aforementioned gap between decentralized nodes and VOs. The Critical Friend Model is comprised of a set of general workflows and algorithms to make better use of node's knowledge of the neighborhood derived from historical collaboration, which is kept in the local storage and known as the metadata snapshot. In addition, a set of related grid components are also introduced to give the visible implementation roadmap of the CFM in the near future.**

***Index Terms*—Inter-cooperative grid architecture; Critical Friend Model (CFM); Metadata snapshots; SmartGRID; Community-Aware Scheduling Protocol (CASP).**

I. Introduction

Job sharing between decentralized distributed heterogenous nodes has been the goal of distributed computing both in academic and industrial fields for decades. The difficulties exist not only because the technical complexity of interoperation betweens nodes, but also policy constraints and boundaries between various institutes and virtual organizations (VOs). In this case, an effective scheduling approach which aims at being able to fit scaled resource community needs to get across the boundaries of different VOs by utilizing unexploited information such as historical collaboration credits and knowledge [1].

The idea of this paper is to use heuristic data to facilitate the scheduling decision making process. Especially, exploiting historical interoperation metadata cached on each grid node would lead to a *demand centered* grid scheduling framework across multiple VOs. As mentioned above, conventional grid VOs are bounded due to various non-common reasons, and realistic job delegations normally only happen between nodes within the same VO. Such approach does not take the full advantage of the fact that a node could belong to more than one VO; especially when job delegation to a node of another VO will not result in a conflict to the original purpose, e.g., critical security issue that only allows job execution within the same VO.

The main contribution of this work is the proposal of a novel scheduling solution named the Critical Friend Model (CFM), which make use of the metadata of each participating node, i.e. the knowledge of neighborhood grid topology and records of previous interaction (either direct or indirect), to generate an empirical scheduling decision across individual nodes from independent VOs. The CFM considers interconnected nodes known via historical realistic collaboration records as the Self-led Critical Friends (SCF) to each other, and together represent a Critical Friendship based Community that has crossed the boundaries of isolated VOs. Furthermore, the strength of the Critical Friendship between nodes is determined by more "subjective and empirical" factors, such as the quantity and quality of previous interactions.

Nodes adopting the Critical Friend Model can work together well since they are following the same philosophy; moreover, to maximize the effectiveness and enable collaboration with other nodes that follow different scheduling solutions and philosophies, the CFM is designed to be easily integrated with a general scheduling guideline entitled Community-Aware Scheduling Protocol (CASP) [2], which has proven to be capable of providing adaptive and effective scheduling solutions in dynamic network [3].

Besides the theoretical model, a set of practical grid components are also considered as important with regard to their help towards the implementation of a Critical Friend Model based distributed computing infrastructure. In this case, an existing project named the SmartGRID is targeted for the future implementation.

The SmartGRID is a cooperative project aiming at increasing the efficiency, robustness, and reliability of heterogeneous grid computing infrastructures [4] concerning volatile and dynamic resources. The proposed grid middleware has been designed as a generic and modular framework supporting intelligent and interoperable grid resource management using swarm intelligence algorithms and multi-type grid scheduling. SmartGRID uses a layered architecture and aims at filling the gap between grid applications, which act as the resource consumers, and the grid resource low-level management systems, which behave as the resource providers. To achieve this goal, SmartGRID uses an autonomic and evolutional grid community composed of its grid schedulers called MaGates [5].

Within the SmartGRID, the discovered information for each specific task is currently discarded after its usage. We aim at extending this model so that each node of a SmartGRID community might also be capable of keeping a metadata snapshot of known remote nodes, in order to facilitate a more efficient and intelligent behavior towards relevant scheduling decisions. Moreover, as the SmartGRID architecture strives to provide intelligent scheduling for the scope of serving the grid community as a whole, not just for a single grid node, the extended work is also concerned with the design of a scheduling strategy supporting the combination of various interoperable bounded grid communities. In this case, we propose to exploit already discovered grid nodes and store metadata snapshots that would facilitate more convenient, efficient and intelligent subsequent resource discovery operations.

The remainder of the paper is organized as follows: related knowledge concerning the trust and correlation on distributed computing is introduced in the next section. The strategy and principle of this work is introduced in Section III firstly, and then detailed in Section IV. A scenario case study is illustrated in Section V, and the work is summarized in Section VI.

II. Trust Related Overview

The Critical Friend Model is established upon the conception of trust in computer sciences.

A. Trust in Computational Environment

[6] defines trust (or symmetrically, distrust) as a particular level of subjective probability with which an agent assesses and monitors that another agent or group of agents is capable to perform and deliver a particular action. The notion of trust as an expectation (as a rational, affective or a mixture of both) is also a closely related concept to that of confidence levels. The idea that the confidence level can be measured is developed by [7] who pointed out that we arrive at the concept of trust by choosing to put ourselves in someone else hands, in that the behaviour of the other determines what we get out of the situation. Others [8] [9] have also stressed the importance of trust building over time (i.e. the temporal dimension of trust). Trust is not only clearly dependant on our past experiences but is also an expectation of reliability and confidence in future events too. Work relevant to the reputation notion has already been carried out by researchers in developing and applying various mathematical formalisms that can be used to design and implement trust models embedded within autonomic systems. Many of them rely on the calculation of local trust thresholds of various kinds. This approach has also been extended so as to seek to enable MAS (Multi-Agent Systems) with the power to investigate trust credentials, provenance and reputation. Confidence levels can also be calculated in various ways. Within the specific context of the consumption and provision of Grid services from VOs previously published work has shown the value of computationally heavyweight confidence engines as exemplified in [10]. There are also works as described in [11] which incorporate high-level proxy measures of VO reputation using purely rational measures derived from previous performance history.

B. Critical Friends and Self-led Trust

The work herein is based upon the notion of critical friends [1] and self-led trust management. The main concept involved is that a community of VO users (encompassing service consumers and providers) can communicate within their own VO network, and manage their own perceptions of other users. In such a system, a user, for example, Alice, can decide (based upon limited local knowledge) how much trust to place in another user, Bob. It is also very important to emphasize that users may belong to more than one VOs and thus, the many-to-many relationship between users with users and users with VOs can further strengthen the level of trust or mistrust. The concept herein is that the VO system is fluid and dynamic, and based upon a series of interactions between users, the trust value between users can evolve over time. The notion of critical friends and self-led trust mirrors the notion of trust relationships in the real world. If a person does not know someone or something, they will ask their friends about it. Based upon the feedback they receive a judgement (personal, and self-led) can be made. Meanwhile, a weight filter mechanism can be introduced upon the degree of usefulness of opinions previously provided by the Critical Friend. Our notion moves away from a centralised trust authority as this could lead to a single point of failure.

III. Strategy and Principle

As originally proposed in [1] and then further discussed in [12], in order to construct a collaborative computing environment across multi-VOs, we propose a novel model entitled *Critical Friend Model (CFM)*. The basic idea of the CFM is to utilize historical collaboration records, which are considered as trust correlation between nodes according to previous experience, to facilitate the job sharing and node cooperation regardless the boundaries of various Virtual Organizations (VOs). Within the Critical Friend Model, the notion of the *Self-led Critical Friend* is introduced as a mean of describing the interaction between nodes in a wider, larger- scale and unknown grid community.

The CFM is partially inspired by an existing grid project named the SmartGRID. The SmartGRID offers a loosely decoupled grid architecture in order to ensure the grid scheduling activities is independent from specific adopted resource discovery services. Two approaches are currently available for job delegation on appropriate remote nodes, i.e. the neighboring nodes list based local search policy and the on-demand community search policy.

However, due to the original targeted scenario of the SmartGRID, the node negotiation and cooperation activities are limited within specific community scope because of nodes of the SmartGRID has to adopted the same Ant-based resource discovery service [4]. It doesn't make good use of the fact that a SmartGRID node could be a member of another VO, e.g., another information system, so that gaps amongst diverse VOs can be bridged somehow. It is then one of contribution of this paper to broaden the node cooperation topology of single VO into multi-VOs. The obtained novel topology comprised of several VOs is entitled as the *Critical Friend Community (CFC)*.

Moreover, due to the knowledge of reachable remote nodes on single participating node is enlarged, job scheduling decisions making process is supposed to be improved because more information from different sources can be obtained compared to the conventional grid, as well as more tasks have to be disposed. In this case, the CFM is comprised of a set of general patterns and algorithms, which are used to organize resources and tasks from both local users and remote collaborators according to specific user preference.

Another noticed defect of traditional usage is that the discovered information for specific task is discarded after usage, although it contains more information rather than a single action, e.g., the trust weight between two independent grid nodes. To overcome such weakness and to make better use of the archived historical information, particular attention is given to maintain aforementioned data and to construct a storage of the so called *Metadata Snapshot* for each involving node. The Metadata Snapshot is a set of collected and evaluated data from various sources, such as resource profile, resource status, and collaboration record archive. By considering such information, future empirical scheduling is believed being able to offer more intelligent decisions due the awareness of the context and kept up-to-date knowledge.

Finally, to enable the cooperation between nodes no matter whether they follow the philosophy of the CFM or not, a high-level scheduling protocol entitled *Community-Aware Scheduling Protocol (CASP)* [2] is to be integrated so that the CFM is able to work together with other scheduling policies despite the extra- facilities.

IV. Implementation

The concerning components of the Critical Friend Model which are introduced in previous section will be detailed as follows:

A. *SmartGRID framework*

Currently, the SmartGRID architecture consists of three parts: the Smart Resource Management Layer (SRML), the Smart Signaling Layer (SSL), and the Data Warehouse Interface (DWI).

The topology of the SmartGRID is an interoperable grid scheduler community composed of engaged decentralized grid schedulers from the SRML, named as MaGates (Magnetic Gateway) [5] and designed to be modular and emphasizing scheduler interoperation. With the infrastructure information retrieved from the DWI, the MaGates discover and connect to each other, so as to collaborate in order to bridge heterogeneous grid systems with a consensual view. The grid community evolves dynamically, and is able to automatically recover from failure situations. Information about available resources and network status is gathered by the SSL, and stored into DWI's distributed data storages. The SSL maintains an overlay network of Nests that provide the runtime environment for the execution of bio-inspired ant algorithms [13] [14]. This approach provides an adaptive and robust signaling mechanism, supporting both grid resource discovery as well as monitoring. The layered architecture of the SmartGRID is shown in Figure 1.

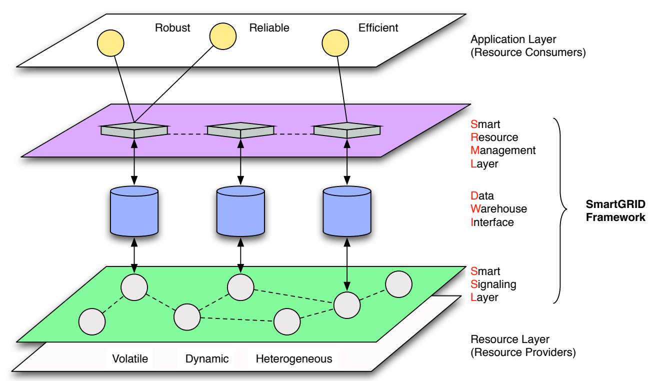


Fig. 1. SmartGRID architecture overview

The SRML is responsible for grid level dynamic scheduling and interoperation that provides grid applications with scheduling decisions on dynamic discovered computing resources. Being the core of SRML, the MaGate is also in charge of propagating resource discovery related tasks to the SSL, analyze the returned results, and decide future operations.

In general, the mission of a grid scheduler is to discover appropriate resources for executing jobs across within a single grid community. The vision of the MaGate scheduler is that of a wider grid community scheduling process is able to exploit resources in large and partially unknown grid communities, and dealing with continuously changing job queues. Thus, since each community node is supposed to receive jobs from both its local and remote grid communities, management of the job queue must deal with a more dynamic, fluid and unexpected environment. The goal is to ensure robustness, reliability, efficiency, and intelligent scheduling response. In this respect, the scheduler must compromise between accepting community jobs and local ones, depending on its workload, *agreement offers* [15], and the ratio between resources contributed to the local and the global grid communities.

The SSL [4] represents the interface from and to the network of the SmartGRID architecture, by providing access to Virtual Organization (VO) resource. The SSL is controlled by the SRML, and provides information about the availability of other resources on other nodes, as well as their status.

From the SSL point of view, each node has some partial knowledge of the underlying logical network. Remote nodes that fall into this partial view are called *direct neighbors*, because they are considered as having good connection with the host node. The SSL hides the complexity and instability of the underlying network by offering reliable services based on distributed ant algorithms. Ant algorithms do not require centralized control, and are known to be robust and adaptable, thus well suited for dynamic networks. Ants are defined as lightweight mobile agents traveling across the network, collecting information on each visited node: a distributed middleware named Solenopsis [16] provides an environment for the execution of ant colony algorithms, in particular the specific designed BlåtAnt collaborative ant algorithm [17]. The activity of the SSL can be either reactive or proactive. Reactive behaviors are controlled by incoming requests from the SRML: information is asynchronously transmitted through a data warehouse interface, and fetched by the local nest. The same interface is used to provide feedback and results on the execution of algorithms. Continuous pro-active activities, such as network monitoring, are used to enhance the QoS of provided services for the SRML, and the robustness of the whole system.

B. Extended Topology

As mentioned before, the current SmartGRID network topology implies that propagated ants searching resources within a specific grid community, which is bounded due to various reasons, such as shared community policy, trust issues, geographical location, etc. Let us now label this bounded grid community as a Virtual Organization 1 (VO_1). In a similar vain, let us assume that there are a number of separated VOs across a wider (larger-scale and thus unknown) grid community (VO_1, ... VO_n). Let us also assume that an individual node is member in more than one VO (e.g., VO_1, VO_2) and that each node within a VO can be a service consumer, service provider or both.

This extended inter-cooperative grid vision, entitled as Critical Friend Community (CFC) and illustrated in Figure 2, enables a network which clearly extends the aforementioned SmartGRID topology. This is mainly due to the fact that the current SmartGRID framework takes job delegation decisions on the basis of ants searching across one and only one VO (e.g., VO_1) and it does not take the full advantage of the fact that a node in a VO_1 can be also a node in a VO_2. In such a grid community, a node, for example, n_1 in VO_1 can communicate with another neighbor node, n_5 in the same VO_1. The rationale is that a node, n_5 can be also a member of another VO_2, which in turn leads to the rationale that n_5 can communicate with another neighbor node such as n_9 (that is also a member of VO_2 and VO_3) and so on. The assumption here is that communicating and/or delegating a job to a node belonging to a different VO will not result to a conflict of interest between parties. Having said that, the assumption is valid given the fact that a VO should not allow membership of a distinct node been part of two conflicting VOs unless it is unknown or there is a certain level of trust. In the case of the latter point, the assumption is still valid given the fact that the associated *agreement offer* and policies explicitly specify the range of act of a job delegation (what is acceptable). On the other hand, if decisions made by two interacting nodes conflict, an agreement [15] based negotiation mechanism [18] can be introduced to address such issues.

The vision is also based upon the very important notion of the *Self-led Critical Friends* (SCF). This concept is built upon relations between nodes, and the knowledge that each node constructs about some neighbor node (either a member of the same or of a different VO) based on previous (direct and/or indirect) interactions, such as communications and delegations. The notion of previous interactions between neighbor nodes determines the strength of the relation and ultimately the level of Critical Friendship. We thus consider a topology of a wider dynamic grid community, based upon a series of SCF relations between nodes from different communities; the strength of a relation between two nodes can be either constant or can evolve over time and influence decisions of the job delegation task. This idea is similar to the one proposed in [19], and effectively creates a second level overlay that can be exploited for efficient resource discovery.

The concept of SCF mirrors the notion of relationships occurring in the real world. If a person (node in our case) is looking for a specific service and they do not know how to find it, they will ask some of their friends (neighbor nodes) who may know it (decision based on past experience). If they do not, these friends will pass the query on to their own friends with the view that someone across the “friends” network (neighbors network in our case) will know and have information relevant to the original request about the specific service. Based upon this information a decision can be made. We purposely moved away from a centralized authority as this could lead to a single point of failure. To explain further, a centralized authority could be compromised by an external entity, and if all users are dependent upon this entity then the functionality of the whole network could fail very quickly. This extended resource discovery topology clearly increases scalability and thus is the most suitable solution for wide grid communities.

C. Local Policies of the Critical Friend Model

The Critical Friend Model (CFM) is able to make decisions based on information from both infrastructure providers and knowledge of critical friends. The CFM is supposed to be understood and carried out by a coordinator component. Taking into consideration that each node within the Critical Friend Community (CFC) has its own local scheduling policies, as well as full control of the local resources, the coordinator is supposed to collaborate with the existing local scheduling policies, and provides a broad view by enabling the participation of remote nodes.

A coordinator is different from a meta-scheduler, although they may be physically the same component sometimes. A meta-scheduler simply assigns a job to local LRM for execution, while two coordinators have to negotiate on a job delegation from one node to the other. That is to say that a

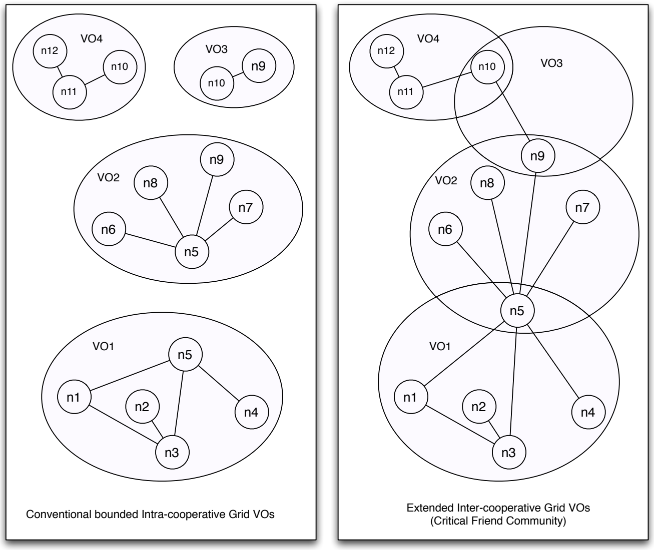


Fig. 2. Vision of Intra-cooperative and Inter-cooperative Grid Topologies (Critical Friend Community)

job delegation request issued by a coordinator can be refused or altered, which is not the case for a meta-scheduler.

If a job delegation request is refused or altered, the initiators coordinator has to continue by either reporting the failure to user, or releasing a re-negotiation process with modified parameters. The approach to automate the above process can be comprehended as a *workflow based* schedule, because the coordinator has already determined steps to do for handling subsequent behaviors like re-negotiation and failure. Regarding the scheduling process of each CFC node concerns many volatile factors retrieved from various environments, adaptability is a critical capability to exploit potential opportunities of fulfilling received job execution requests without bothering the initiator user.

The Critical Friend Model is comprised of two behavior patterns, namely the *Job Arrival Pattern* and the *Job Complete Pattern*.

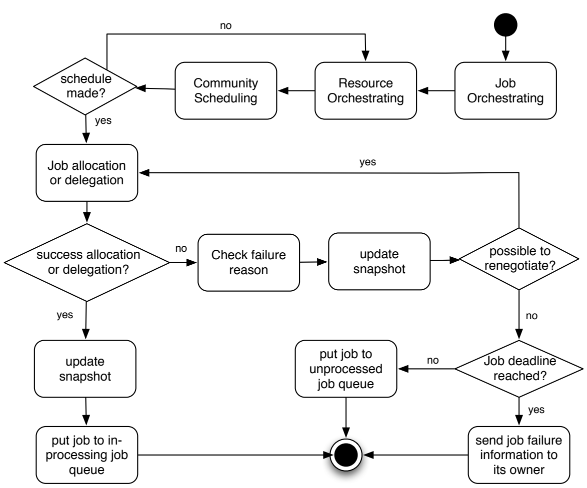


Fig. 3. Critical Friend Model Job Arrival Pattern

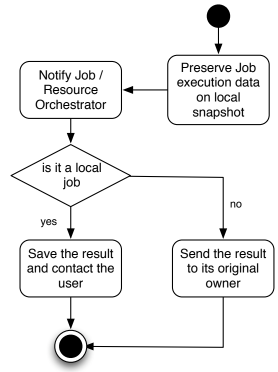


Fig. 4. Critical Friend Model Job Complete Pattern

The Job Arrival Pattern, as illustrated in Figure 3, starts from the *Job Orchestrator*, a component responsible for generating next to-process job depending on continuously arriving incoming jobs, no matter where they come from (either the local node or a remote one). Conversely, the Job Complete Pattern, as illustrated in Figure 4, is triggered by notification events of newly accomplished jobs, from either local resource or remote nodes, in order to preserve useful data retrieved from job execution (e.g., job SLA satisfaction [20], used resource CF weight re-calculation, etc.) for future use.

Besides, the CFM is supposed to be complemented by several local policy relevant orchestration algorithms, such as the simplified Job Orchestrating Algorithm, Resource Orchestrating Algorithm, and Community Scheduling Algorithm, which will be discussed below.

An interaction mechanism between involving coordinators is a critical component for the design of the CFM. Such a mechanism should be flexible to adapt to different scenarios, platform independent to decouple the participators from the infrastructure, robust to recover from an unavailable/failed agreement, and automated to handle continuous incoming requests.

More specifically, the implementations of the aforementioned algorithms are detailed as follows:

1) Job Orchestrating Algorithm (JOA): The philosophy of JOA is to organize a to-process job queue by merging diverse job incoming sources, with respect to local user preference.

If the preference indicates that local jobs have higher priority, the JOA will try to fill the size limited output queue with jobs from local queue firstly, and pick appropriate jobs from other sources, e.g., community queue or unprocessed queue, only if the limit of the output queue is not exceeded. If the user desires an equal treatment for all incoming job requests, the output queue will be comprised of the earliest arrived jobs, no matter where they come from. Finally, if a profitable philosophy is determined, each arrived job will be evaluated, in order to determine individual *job-profite-rate* value. In this case, the output queue will be composed by the most profitable jobs. Once the output queue is generated, the local policy of the participating node is invoked for future processing.

Furthermore, the JOA can be extended by users self-defined job orchestrating policies, and other locally adopted scheduling algorithms, besides herein mentioned FCFS and EasyBackfilling.

Algorithm 1 Job Orchestrating

Require: a local job queue lj, a community job queue cj, a unprocessed job queue uj, a output job queue oj, allowed size of oj $limitoj$, current size of oj $sizeoj$
Require: $jpr = jp/(jtc * jcpu)$
jpr: job profit rate; jp: job profit
jtc: time cost of a job; $jcpu$: required number of cpu of a job

$local.scheduler \in \{FCFS, EasyBackfilling\}$
$user.preference \in \{LocalJobPriority, CommunityJobFair, Profitable\}$
update data of lj, cj, uj
if $user.preference = LocalJobPriority$ **then**
  fill oj with $(limitoj)$ jobs from lj
  if $limitoj$ is not reached **then**
    find $(limitoj - sizeoj)$ jobs from cj and uj to fill oj
  end if
else if $user.preference = CommunityJobFair$ **then**
  fill oj by selecting $(limitoj)$ jobs from lj, cj and uj depending on arrival time fairly
else if $user.preference = Profitable$ **then**
  for job from queue lj, cj, uj **do**
    if jpr of job is not determined **then**
      calculate jpr for job
    end if
  end for
  fill oj by selecting $(limitoj)$ jobs from lj, cj and uj, depending on jpr fairly
else
  generate oj with random picked jobs
end if
determine the validate time of oj
output oj
invoke $local.scheduler$ to execute oj

2) Resource Orchestrating Algorithm (ROA): The ROA is responsible for generating a set of appropriate candidate resources for each input job request, depending on user preference.

If the *LocalResourcePriority* policy is chosen, resources owned by the local node are considered firstly, with an additional selection from other list (e.g., community resource list and critical friend resource list) only occurring if the size limit of the output resource list is not achieved. If the policy *CommunityResourceFair* is preferred, a fair selection is carried out on all known resource list. Finally, if the policy *FriendResourcePrioriy* is specified, the output list will firstly pick up a suitable resource owned either locally or by some critical friends, with other list not being considered unless the output resource list is not full.

Similarly to the aforementioned JOA, the ROA can be extended by user self-defined resource orchestration policies, but only if the expected known resource list can be found within the local node's snapshot storage.

Algorithm 2 Resource Orchestrating

Require: a local resource list lr, a community resource list cr, a critical friend resource list fr, a output resource list or, limit of output resource list $limitor$
Require: incoming job requirement jr

$user.preference \in \{LocalResourcePriority, CommunityResourceFair, FriendResourcePriority\}$
update data of lr, cr, fr
if $user.preference = LocalResourcePriority$ **then**
  fill or with selected resources from lr firstly
  if $limitor$ is not reach **then**
    fill or with fairly selected resources from cr and fr
  end if
else if $user.preference = CommunityResourceFair$ **then**
  fill or with fairly selected resources from lr, cr and fr
else if $user.preference = FriendResourcePriority$ **then**
  fill or with fairly selected resources from lr, fr
  if $limitor$ is not reach **then**
    fill or with selected resource from cr
  end if
else
  generate or with random selected appropriate resources
end if
output or

3) Community Scheduling Algorithm (CSA): Once a candidate schedule (a job with its candidate resource list) arrives, an allowed maximum scheduling time duration will be given to prevent unacceptable delays and performance loss. The CSA is responsible for contacting the candidate resources simultaneously within allowed delay, in order to get a job allocation/delegation *agreement* [15] based on the expected request (in our case, it is an *agreement offer*). An *agreement* means that the job execution request is approved by the target resource (either locally or remotely) and if such job can be delivered within a certain time, it will be accepted and executed under the agreed terms. As soon as an *agreement* has been made between the requesting node and a target resource, other *agreement offers* will be revoked.

In case no candidate resources are able to accept such *agreement offer* due to various reasons, e.g., local workload, local policy alternation, latest resource status change, the CSA needs to check whether the allocated scheduling time has expired. If not, the CSA is able to contact the locally adopted Information System, and asks for a live search from the located VO within the remaining scheduling duration. If appropriate resources can be found within such time constraints, a parallel (re-)negotiation with a newly prepared *agreement offer* can be issued again, within the shortened time duration.

As mentioned, although the job allocation is a different operation from job delegation (because the targeted resource of job allocation is an owned LRM of the local node, which cannot negotiate a job acceptance), the CSA is not concerned with such slight difference and ignores the *agreement offer* based (re-)negotiation process if the target resource is managed by a local LRM.

Algorithm 3 Community Scheduling Algorithm

Require: prepared candidate resource list cr
Information System of the located VO IS
current processing job job
allowed scheduling time ts, current time t, expected time deadline $tstop = (t + ts)$
agreement offer on job allocation/delegation $offer$
agreement on job allocation/delegation $agreement$

get the allowed scheduling time ts for job
repeat {parellel}
  negotiate/re-negotiate $offer$ within ts
  if $agreement$ made **then**
    break
  end if
until each resource of cr is contacted
if $agreement$ not available **then**
  if $t < tstop$ **then**
    invoke IS within $(tstop - t)$
    repeat {parellel}
      negotiate/re-negotiate $offer$ within $(tstop - t)$
      if $agreement$ made **then**
        break
      end if
    until each discovered result of IS is contacted
  end if
end if
if $agreement$ not broken and $agreement$ not expired **then**
  allocate/delegate job based on $agreement$
end if

D. Metadata Snapshots

Metadata Snapshots help to address the issue of leading to intelligent and empirical future scheduling decisions according to previous job sharing experience.

As mentioned earlier, every node within the grid community must publish its capabilities; moreover, the assumption is that this public profile is kept continuously up-to-date. Specifically, within the framework of SmartGRID, each MaGate scheduler generates directives to propagate ants in the SSL in order to provide nodes with availability and status information. Current MaGate resource discovery service either utilizes partial knowledge stored in a node's cache, or delivers search queries to match individual job delegation requirements with published node profiles from the known bounded grid community. To achieve this, the SSL is employed to hide the complexity and instability of the underlying network by utilizing ant algorithms. These ants function as lightweight mobile agents traveling across the grid network, collecting information on each visited node.

The herein objective is to extend current SmartGRID functionality by enabling the MaGate resource discovery service to utilize more knowledge that can be made available from the visited nodes. In this respect, we propose that each time a node, n_1, delegates a job to another node (with no regard to the possibility that it might be a member in multiple VOs) such as node n_3, node n_1 is required to keep an instance profile with regard to the parameters which have been used to discover it as a resource originally, as well as, the quality of service provided by node n_3. In a similar way, it is expected that every time a node, n_1, delegates a job to n_3, node n_1 has to update the profile about n_3 in its cache. We suggest that this is a bi-directional commodity and thus, we expect that node, n_3 will also keep an instance profile about n_1 in its cache. We also assume that a node n_1 stores as many profiles in its cache as the number of previously visited nodes. The vision is to enable nodes storing meaningful information that can help assist them and their critical friends at a later stage.

Apparently, these profiles residing in each cooperating node can be sustained or even evolve over time and act as the tool towards decision making for delegating a job next time. That is, a calculation as an aggregative and weighted value representing the (strength or else) critical friendship relationship between nodes that previously cooperated, could significantly improve the decision making towards a job delegation to a particular node (or cluster of nodes) that is available from a pool of discovered resources across the wider grid community. Such a notion clearly provides a richness not seen in any other resource discovery or job delegation model.

As mentioned in subsection IV-C, regarding many factors that could impact the scheduling decision, each node of the Critical Friend Community (CFC) has a metadata snapshot. Each metadata snapshot is comprised of sets of schemes. A scheme is a group of elements that is used for describing a particular resource or purpose. For example, a machine scheme is normally composed of elements such as machine architecture, operating system, number of CPU, etc. Other important schemes include: local resource profile, local resource status, agreement offer list, known Self-led Critical Friend (SCF) profile list, known SCF recent status list, historical processing records, etc. Noteworthy, data stored within each scheme is kept up-to-date over time, and is being evaluated and weighted to facilitate intelligent scheduling for the future incoming job requests.

To remedy the pain of organizing all necessary information (static and dynamic) of a Self-led Critical Friend (SCF), as well as to represent such knowledge is easy-to-understand way, the notion of *Snapshot Profile* is proposed. As shown in Figure 5, A *Snapshot Daemon* is responsible for gathering metadata distributed in different schemes, and representing the knowledge of each individual SCF in a clean and well-organized way. The Snapshot Profile concerns all valuable knowledge of a critical friend, including: CF location, configuration, static and dynamic status, installed application list, tariff, weight (as a CF of the local node), historical SLA (depending on job type) [20] [21], historical charge-load arrange (depending on job type), prerequisite (depending on job type), time-stamp (indicating until when this information can be considered as up-to-date).

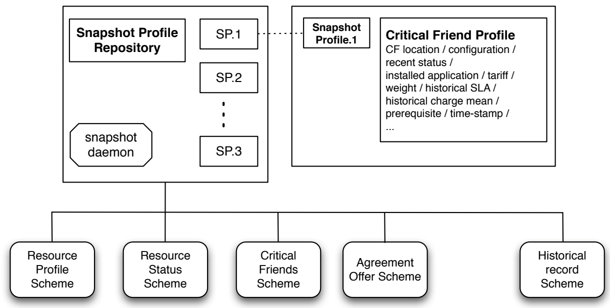


Fig. 5. Metadata Snapshot Structure

Finally, the *Snapshot Daemon* is also responsible of handling metadata exchange, either proactively or reactively, with other nodes of the CFC.

E. Community-Aware Scheduling Protocol

Besides the Critical Friend Model itself, to enable nodes following the principle of the CFM are able to collaborate with nodes with different philosophies, the integration with a general scheduling protocol, namely the Community-Aware Scheduling Protocol [2], is expected.

The basic idea of the CASP is to make scheduling decisions upon grid community, instead of isolated job queue on each grid node. The protocol dedicates to spread the all involving scheduling events, e.g., job submission and job queue optimization, across the network in order to reach as many candidate nodes as possible. The CASP is comprised of two main phases:

1) Job Submission Phase: Grid users can submit jobs to any nodes of the grid community. The initial job recipient, referred as the *initiator*, issues a resource discovery across the grid overlay by broadcasting a REQUEST message. The *initiator* then waits for a predefined timelapse for incoming query replies. A node receiving a REQUEST messages checks if the required profile matches its own capability. Accordingly, it computes an estimated completion time/cost based on actual resources and current scheduling, and forwards this information by means of an ACCEPT message.

The *initiator* then evaluates incoming ACCEPT responses, and selects the best suited node (i.e. the node providing the least time to completion or lowest execution fee). The latter is assigned the job by means of an ASSIGN message, and is referred to as the *assignee*.

2) Scheduling and Rescheduling Phase: The *assignee* is responsible to manage and execute the assigned job according to its own scheduling mechanism and policy. Based on the *initiator*'s offer selection mechanism, the *assignee* is the node that provides the shortest time-to-completion or lowest cost for that particular job. However, availability of resource on the grid and scheduling of jobs may change due to various reasons, such as new nodes connecting to the grid or existing job cancelation. Thus, the *assignee* may not remain the best solution.

In this case, the *assignee* may generate a number of INFORM messages as far as the job execution has not yet started, and send them over the network using a low-overhead random walking protocol. INFORM messages' content is similar to REQUEST messages, but their purpose is to inform other nodes about the job's current schedule on the *assignee* node. A node receiving an INFORM message checks if it matches its up-to-date profile [22] and evaluates an estimated value (e.g. completion time) according to its own schedule. If such estimation leads to a better result than the INFORM message, the node will send an ACCEPT message to the node that currently manages the job. The current *assignee* receiving the message may then choose to re-assign the job by means of an ASSIGN message.

Regarding the CASP only suggests a set of general rules towards a collaborative scheduling decision making process, it can cooperate with different local scheduling policies and mechanisms, including the Critical Friend Model.

V. CASE STUDY

We are interested here in describing the proposed novel inter-cooperative process and related events sequence in order to illustrate the behavior of the Critical Friend Model. In the context of the SmartGRID, this involves ants as agents acting on behalf of neighboring nodes in order to enable the MaGate scheduler to discover, decide and assign job delegations on suitable resources. To achieve this, we are using the aggregative case scenario (ACS) below. Figure 6 illustrates the low-level events transferring workflow.

ACS: Let us assume that a VO_1, consists of 8 nodes (nodes n_1 ... n_8). Let us assume that a VO_2, consists of 6 nodes (nodes n_7, n_9, ... n_{13}). Let us also assume that a node, n_1 in VO_1 wishes to delegate a job to another node $n_?$, using ants a_2, a_4 and a_7, which are propagated according to queries issued by the MaGate resource discovery service. The following sequential steps describe this novel, inter-cooperative process:

1) Node n_1 in VO_1 invokes its ants a_2, a_4 and a_7;
2) Ant a_2 contacts the neighboring node n_2 (that is a critical friend: $cf_{1,2}$), ant a_4 contacts the neighboring node n_4 (that is a critical friend: $cf_{1,4}$) and ant a_7 contacts the neighboring node n_7 (that is a critical friend: $cf_{1,7}$. It's noteworthy that node n_7 is a member of both VO_1 and VO_2);
3) Ant a_2 reads and collects the public availability profile, as well as its metadata snapshots about previous job delegation activities completed in node n_2 that are available from the cache of node n_2;
4) With discovered metadata snapshots (available from the cache of node n_2) by ant a_2, MaGate n_1 realizes that node n_3 is a $cf_{3,2}$; moreover, n_3 has the capacity to take the job delegation task jd_1;
5) Ant a_4 reads and collects the public availability profile of n_4, as well as its metadata snapshots about previous job delegation activities completed in node n_4 that are available from the cache of node n_4;

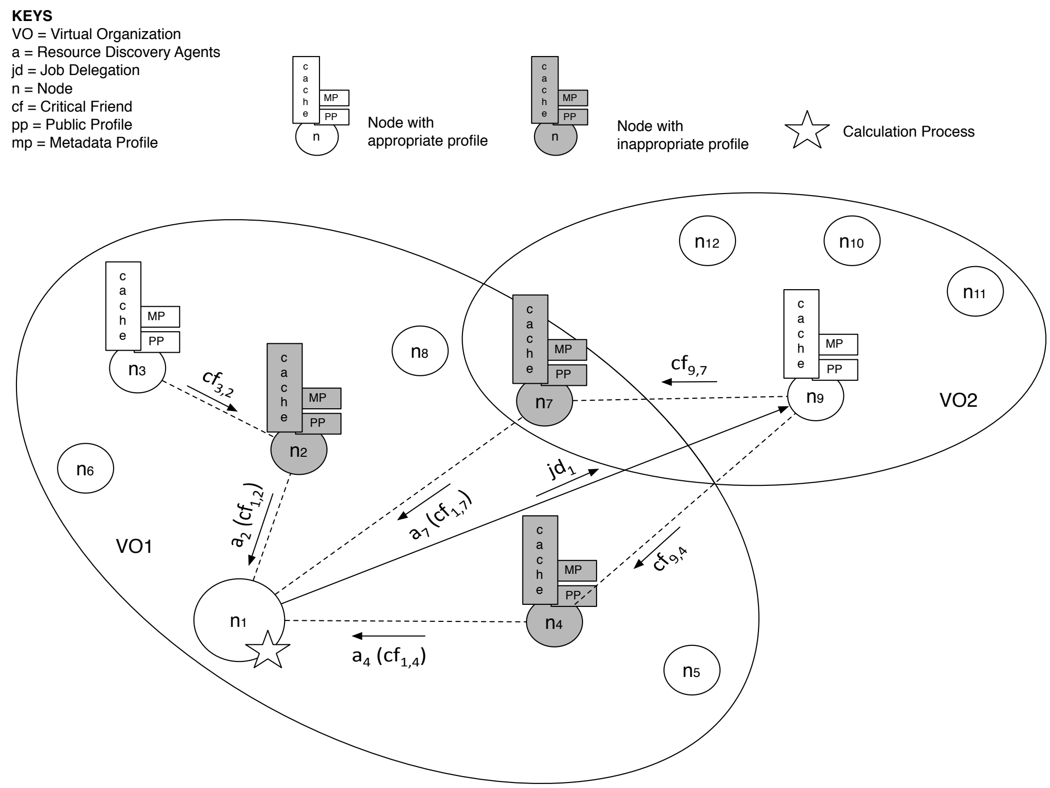


Fig. 6. SmartGRID based Critical Friend Model collaboration workflow

6) With discovered public availability profile by ant a_4, MaGate n_1 realizes that node n_4 has no capacity to take the job delegation task jd_1;
7) With discovered metadata snapshots (available from node n_4 cache) by ant a_4, MaGate n_1 realizes that node n_9 is a $cf_{9,4}$; moreover, n_9 has the capacity to take the job delegation task, jd_1;
8) Ant a_7 reads and collects the public availability profile of n_7, as well as its metadata snapshots about previous job delegation activities completed in node n_7 that are available from the cache of node n_7;
9) With discovered metadata snapshots (available from the cache of node n_7) by ant a_7, MaGate n_1 realizes that node n_9 is a $cf_{9,7}$; moreover, n_9 has the capacity to take the job delegation task jd_1;
10) Ants a_2, a_4 and a_7 collect profiles about nodes n_3 and n_9 for MaGate n_1, which reports such information to a virtualized data warehouse;
11) We now assume that a calculation as an aggregative and weighted value representing the (strength or else) critical friendship relationship between previously cooperating nodes, $cf_{3,2}$, $cf_{9,4}$ and $cf_{7,9}$ has significantly improved the decision making towards jd_1 to node n_9. This is due to the aggregative cf values that have suggested that node n_3 has been delegated x number of jobs and the satisfaction (confidence) level was significantly less than the satisfaction (confidence) level provided for an equal number of past delegated jobs in node n_9.

VI. Conclusion

In order to go beyond the boundaries of the conventional grid virtual organizations (VOs), a novel cooperative mechanism entitled the Critical Friend Model (CFM) has been proposed in this work. The kernel conception of the CFM is the Self-led Critical Friends (SCF) maintained on each participating grid node which stands for a set of remote grid nodes known by the host node and weighted according to historical collaboration records and experience. The SCF are trusted due to their previous behaviors during the collaboration with the host node, and can be contacted without regard to adopted scheduling policies, resource discovery limitation, or VO boundaries.

To demonstrate a doable roadmap, the principle, as well as theoretical components of the CFM are illustrated. The CFM presents an extended grid topology inspired by an existing grid project named the SmartGRID. The SmartGRID has been developed to be a generic and modular framework to support intelligent and interoperable grid resource management using swarm intelligence algorithms. The CFM addresses how grid schedulers from various bounded grid communities could be used in a manner that would extend current SmartGRID functionality.

In the context of the CFM, a novel cooperative mechanism that makes use of historical collaboration experience is introduced. More specifically, the CFM consists of several general patterns, e.g., the Job Arrival Pattern and the Job Complete Pattern, as well as a set of algorithms, e.g., the Job

Orchestrating Algorithm (JOA), the Resource Orchestrating Algorithm (ROA), and the Community Scheduling Algorithm (CSA), in order to serve future scheduling and collaboration behaviors well with obtained metadata within the scope of extended topology. Aforementioned patterns and algorithms can have customized preference in order to fit various local requests and constraints.

To preserve the historical collaboration records within the decentralized distributed grid nodes, a metadata snapshot is required on each participating node. The metadata on each node is established from various sources, e.g., resource profile, resource status, adopted resource discovery service cache, and node coordinator archive. Data is re-organized within the metadata snapshot into diverse Snapshot Profiles, so that they can be easily searched and analyzed.

In order to enable the interaction between nodes following the CFM philosophy and nodes which don't follow, the Community-Aware Scheduling Protocol (CASP) is proposed to be integrated. CASP is mainly comprised of two operating phases to disseminate particular important scheduling events, e.g., community job arrival, first scheduling within limited duration, and re-scheduling for nodes with long waiting time. Regarding no preference on the local facilities and local policies, CASP is able to fulfill the integration purpose and allows a transparent and non-mandatory using manner of the CFM.

The reference experiment of the CFM, as well as the complementary components, are being implemented by the responsible research groups [23] [24] [25].

Acknowledgements

This work is financially supported by the Swiss Hasler Foundation [26], in the framework of the ManCom Initiative (ManCom for Managing Complexity of Information and Communication Systems), project Nr. 2122.

References

[1] Y. Huang, N. Bessis, A. Brocco, P. Kuonen, M. Courant, and B. Hirsbrunner. Using Metadata Snapshots for Extending Ant-based Resource Discovery Service in Inter-cooperative Grid Communities. In *International Conference on Evolving Internet*, pages 89–94, Cannes, French Riviera, France, 2009. INTERNET 2009, IEEE Computer Society.

[2] Y. Huang, A. Brocco, N. Bessis, P. Kuonen, and B. Hirsbrunner. Community-Aware Scheduling Protocol for Grids. In *24th IEEE International Conference on Advanced Information Networking and Applications*, pages 334–341, Perth, Australia, 2010. AINA 2010, IEEE.

[3] A. Brocco, A. Malatras, Y. Huang, and B. Hirsbrunner. ARiA: A Protocol for Dynamic Fully Distributed Grid Meta-Scheduling. In *30th IEEE International Conference on Distributed Computing Systems*, Genoa, Italy, 2010. ICDCS 2010, IEEE. To appear.

[4] Y. Huang, A. Brocco, P. Kuonen, M. Courant, and B. Hirsbrunner. SmartGRID: A Fully Decentralized Grid Scheduling Framework Supported by Swarm Intelligence. In *Seventh International Conference on Grid and Cooperative Computing, 2008. GCC '08*, pages 160–168, China, 2008. IEEE Computer Society.

[5] Y. Huang, A. Brocco, M. Courant, B. Hirsbrunner, and P. Kuonen. MaGate: an interoperable, decentralized and modular high-level grid scheduler. *International Journal of Distributed Systems and Technologies (IJDST)*, 2010.

[6] D. Gambetta et al. *Trust: Making and breaking cooperative relations*. Basil Blackwell New York, 1990.

[7] S.P. Marsh. *Formalising trust as a computational concept*. Citeseer, 1994.

[8] S. Dayal, H. Landesberg, and M. Zeisser. How to build trust online: Marketers can build mutually valuable relationships with customers through a trust-based collaboration process. *Marketing Management*, 8:64–66, 1999.

[9] P. Nikander and K. Karvonen. Users and trust in cyberspace. *Lecture Notes in Computer Science*, pages 24–35, 2001.

[10] M. Wilson, A. Arenas, and L. Schubert. Trustcom framework v4. *TrustCoM Deliverable D*, 63, 2006.

[11] S. Song, K. Hwang, and Y.K. Kwok. Trusted grid computing with security binding and trust integration. *Journal of Grid computing*, 3(1):53–73, 2005.

[12] Y. Huang, N. Bessis, A. Brocco, S. Sotiriadis, M. Courant, P. Kuonen, and B. Hirsbrunner. Towards an integrated vision across inter-cooperative grid virtual organizations. In *Future Generation Information Technology*, pages 120–128, Jeju Island, Korea, 2009. FGIT 2009, Springer LNCS.

[13] E. Bonabeau, M. Dorigo, and G. Theraulaz. *Swarm intelligence: from natural to artificial systems*. Oxford University Press, USA, 1999.

[14] M. Dorigo, M. Birattari, and T. Stutzle. Ant colony optimization. *IEEE Computational Intelligence Magazine*, 1(4):28–39, 2006.

[15] A. Andrieux, K. Czajkowski, A. Dan, K. Keahey, H. Ludwig, J. Pruyne, J. Rofrano, S. Tuecke, and M. Xu. Web Services Agreement Specification (WS-Agreement). Technical report, Open Grid Forum, USA, 2004.

[16] Amos Brocco, Béat Hirsbrunner, and Michèle Courant. Solenopsis: A framework for the development of ant algorithms. In *Swarm Intelligence Symposium*, pages 316–323, Honolulu, Hawaii, April 2007. SIS, IEEE.

[17] Amos Brocco, Fulvio Frapolli, and Beat Hirsbrunner. Bounded diameter overlay construction: A self organized approach. In *IEEE Swarm Intelligence Symposium*, Nashville, Tennessee, USA, April 2009. SIS 2009, IEEE.

[18] DGA Mobach, BJ Overeinder, and FMT Brazier. A WS-Agreement based resource negotiation framework for mobile agents. *Scalable Computing: Practice and Experience*, 7(1):23–36, 2006.

[19] Vicent Cholvi, Pascal Felber, and Ernst Biersack. Efficient search in unstructured peer-to-peer networks. In *SPAA '04: Proceedings of the sixteenth annual ACM symposium on Parallelism in algorithms and architectures*, pages 271–272, New York, NY, USA, 2004. ACM.

[20] A.N. Hiles. Service level agreements. *The TQM Magazine*, 6(2):14–16, 1994.

[21] R. Ranjan, A. Harwood, R. Buyya, and A. Victoria. SLA-based coordinated superscheduling scheme for computational Grids. In *IEEE International Conference on Cluster Computing (Cluster 2006), Barcelona, Spain*, pages 1–8. Citeseer, 2006.

[22] Nik Bessis. *Grid Technology for Maximizing Collaborative Decision Management and Support: Advancing Effective Virtual Organizations*, chapter Model Architecture for a User Tailored Data Push Service in Data Grids. IGI, 2009.

[23] Pervasive Artificial Intelligence Research Group, University of Fribourg, Switzerland. http://diuf.unifr.ch/pai, 2010.

[24] GridGroup, University of Applied Sciences Western Switzerland, Fribourg. http://gridgroup.hefr.ch, 2010.

[25] Centre for Research in DIstributed Technologies, University of Bedfordshire, UK. http://www.beds.ac.uk, 2010.

[26] Hasler Fundation. http://www.haslerstiftung.ch/, 2010.

Probe Framework - A Generic Approach for System Instrumentation

Markku Pollari
Technical Research Centre of Finland, VTT
Oulu, Finland
Email: markku.pollari@vtt.fi

Teemu Kanstrén*
Technical Research Centre of Finland, VTT
Oulu, Finland
Email: teemu.kanstren@vtt.fi

***Abstract*—This paper introduces a general framework directed for system instrumentation. The introduced framework provides support for a system instrumentation approach that enables designing information capture, monitoring and analysis features into a software-intensive system. We describe the general concept, architecture and implementation of the framework and two case studies in its application. As a prototyping platform, we dealt with collecting information from Linux systems by probes created with the building blocks and interfaces provided by the framework. We also discuss the effects of building support for the framework in an implementation from the viewpoint of different constraints, especially focusing on real-time embedded systems, where especially strict constraints are present. Overall, we demonstrate the feasibility of a more uniform instrumentation approach through this concept and its application in two case studies.**

***Index Terms*—system instrumentation; monitoring; analysis; testing.**

I. Introduction

This paper extends the work presented in [1], by offering a broader view on the background concepts, and the implementation of the framework and a more detailed account on the experiences and lessons learned during the work.

Understanding and analysing the behaviour of complex, software-intensive systems is important in many phases of their life cycle, including testing, debugging, diagnosis and optimization. In addition to these, many systems themselves are built for the sole purpose of monitoring their environmental data and reacting to relevant changes, such as detecting patterns in internet traffic or adapting to available resources. All these activities require the ability to collect information from the different parts of the system.

These basic activities and requirements in software engineering have existed since the first days of writing software. However, despite this there has been little research and activity to build support for systematic monitoring and information capture into software platforms. Instead, what is most common is the use of ad-hoc solutions to capture data where needed, as needed. In these cases, the instrumentation required to capture the information is added momentarily into the system and removed after the short-term need has passed. Recent studies have still emphasized this problem, describing large-scale systems where these types of features are important but support for them is lacking [2].

In this paper we present a design concept, and its implementation and validation, for a platform to support the systematic capture and analysis of information related to the behaviour of a system and its environment. This platform is termed as the Probe Framework (PF). The PF provides support for building monitoring functionality for collecting information on the behaviour of software intensive systems and using this information for purposes such as built-in features in the software itself (as product features) and testing analysis of the systems during their development (testing and debugging) and deployment (diagnostics). The prototype implementation of PF is available as open source[3]. We also discuss the effects of building support for this type of a framework into a system from the point of view of the system constraints on available resources and effects on its behaviour and performance, especially focusing on real-time embedded systems, where especially strict constraints are present.

This paper is structured as follows. Section 2 discusses the background and motivation for the work. Section 3 describes the main concepts of the PF at a higher level. Section 4 covers the implementation of the PF and Section 5 presents two case studies of utilizing the PF and experiences gained from this work. Section 6 discusses the related work, and finally the conclusions in Section 7 end the paper.

II. Background and Motivation

The concept of capturing information from a system and its environment is often described as tracing the system. Similarly, in this paper we use the term tracing to describe the activity of capturing information from a running system, either with external monitoring or internal instrumentation features. The data captured is described as a trace of program execution. Many different domains make use of tracing information, such are: system security analysis, internet monitoring and protection [4], run-time adaptation and diagnosis[5], optimization [6], testing and debugging [5], [7], [8], [9].

There are various tools around for specific instrumentation and tracing on different platforms, such as DTrace for Solaris [10] and OSX [11], Linux Trace Toolkit Next Generation

*Also affiliated at Delft University of Technology, Faculty of Electrical Engineering, Mathematics and Computer Science, Mekelweg 4, 2628 CD Delft, The Netherlands.

(LTTNG) [12] and SystemTap [13] on Linux. These tools all share a common goal to observe and store traces of system behaviour and resource use, such as CPU load, network traffic and filesystem activity. They are typically intended to provide a trace facility for the low-level resources and related behaviour of the system kernel, using solutions such as their own programming/scripting language to define where to exactly insert trace code into the operating system kernel [10][13]. For example, in our implementation of PF we make use of SystemTap, which allows one to add trace code into the Linux kernel without the need to recompile or reboot the running system [14].

These low-level frameworks provide an excellent basis for capturing low-level information from a system when ad-hoc instrumentation needs arise. However, while these tools are useful for many purposes, they alone are not sufficient for efficient observation of complex system behaviour. Additional useful information from this low-level information can be gained by using advanced analysis methods such as multivariate analysis to infer additional information such as relations and similar properties from the low-level data [15]. However, what is also needed is information on a higher level, including the events, messages and interactions of different parts of the system. Also, information about the environments of these parts and their relation to the lower level details are needed for a complete view of system behaviour.

This type of information is a part of the higher level design of a system, and it is implemented as higher-level abstractions inside the components. Thus, it is not possible to build generic components that would capture this information from all the components from the OS kernel or any custom application. When solutions such as component based middleware are used, it is possible to build part of this support into the middleware itself to capture the data [2]. However, for an effective and descriptive trace, application specific tracing is also needed. For this level of tracing several frameworks exist, such as Log4J [16] and syslog [17]. Additionally, when the availability of such features and information is highly valued, customized support for these have been built into the system as first-class features [2].

The above descriptions show how effective analysis of software intensive systems requires many different types of traces to be supported, collected and analysed together. Different tools need to be used effectively in different steps and finally combined as one for both built-in features and external analysis. Only in this way is it possible to provide the required support to get a definite view of the behaviour of a complex system.

From this viewpoint of complete system analysis and its support through the life cycle, the described trace tools and frameworks suffer from a set of issues. The tools use their own interfaces, custom data formats and storage mechanisms. Additionally, often the storage is only considered in the form of a local filesystem with the intention of being manually exported to external analysis tools or read as such by humans. This can be problematic in different systems and environemnts. For example, simply accessing this information from an embedded system can be very difficult as these systems are often limited in their external interfaces. Even where this is possible, in the case of a deployed system, it is not always cost-effective for someone to go to the field site to examine the trace file. Additionally, the trend for relying on ad-hoc temporary tracing solutions makes it very difficult to capture a meaningful trace of a system as there is no built-in support to be used when needed. The lack of design support for proper tracing from the beginning further brings problems such as probe effects, where addition of temporary trace mechanisms changes the timings and resource usage of the actual running system that is to be analysed [18]. The probe effect is illustrated in Figure 1, by showing how the timing of two tasks is changed by the tracing.

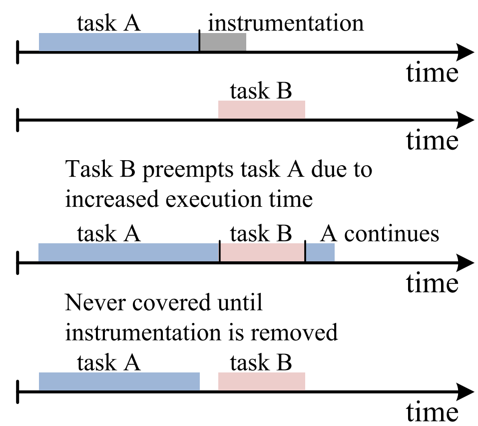


Fig. 1. Probe effect

To address these issues, to build a basis for effective system level tracing, analysis and related program functions, we have developed a trace platform called Probe Framework (PF). Our prototype implementation is created on Linux and enables the collection of trace information both from kernel and user space probes, through a single unified component in the system. By starting with the goal of building these features into the system as first-class features we make it possible to address properties such as probe effects, information access, limited resources and real-time requirements. With a commonly shared and customizable format for the collected trace, it is possible to store and export this information to different analysis tools. With unified interfaces inside the platform it is also possible to easily design built-in features that make use of information from all the various tools. As the main intent is to build a higher-level abstraction mechanism, we use existing tools such as SystemTap and integrate these to the Probe Framework. The PF and its main concepts are described in more detail in the following sections.

III. General concept

On a higher level, the Probe Framework is a part of a larger concept, which includes three main components; trace capture, trace storage database and trace analysis. The PF provides the needed support as a platform to capture the trace information from the system under test. An information database server is used to collect the trace information and provide the means to query, filter and export the trace to analysis tools. Various trace analysis tools can be used to analyse the information provided. This includes tools specifically for trace analysis and also tools more generally intended for analysis of data, such as multivariate analysis. For example, experiences on using a multivariate analysis tool to analyse the network functionality and behaviour of a system from information that can be captured with the help of PF have been studied in [15]. In addition to making use of the captured information in external analysis tools, it is also possible to make use of it as part of built-in product features for processes such as adaptation, testing and analysis. This overall architecture is described in Figure 2.

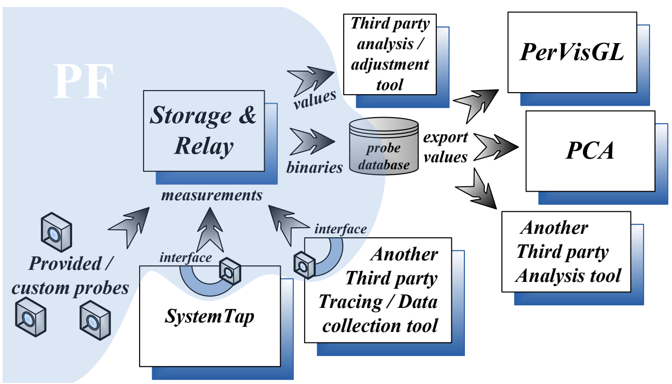


Fig. 2. High level overview of PF and external components.

The Probe Framework itself has a layered architecture as presented by Buschmann et al. [19]. The PF's architecture is divided in three main layers: basic services, monitoring services and test services. Each of these layers builds on the functionality of the layers below it, as described in Figure 3.

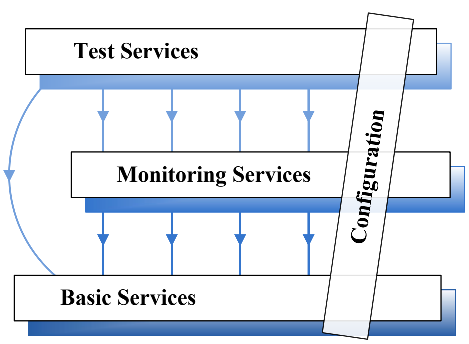


Fig. 3. PF internal architecture.

The basic services contain services deemed necessary for information handling, such as data buffering, storage and relaying to external database. The basic services are general for all the probes, and offer the support for fast implementation of the upper level services. The term probe, in the context of this paper, means the entity that is formed by utilizing the different service layers to create the functionality for collecting and handling the monitoring of some aspect of the target system. The basic services comprises of three parts; the first part is the probe interfaces, the second is the binary formatter and the third is the communication handler. These three parts are illustrated in Figure 4, the figure also describes the internal division of the basic services. It can be seen that the binary formatter and the communication handler are encapsulated as a storage and relay (module), but without going into the details here the upcoming Section IV-B offers more on this subject.

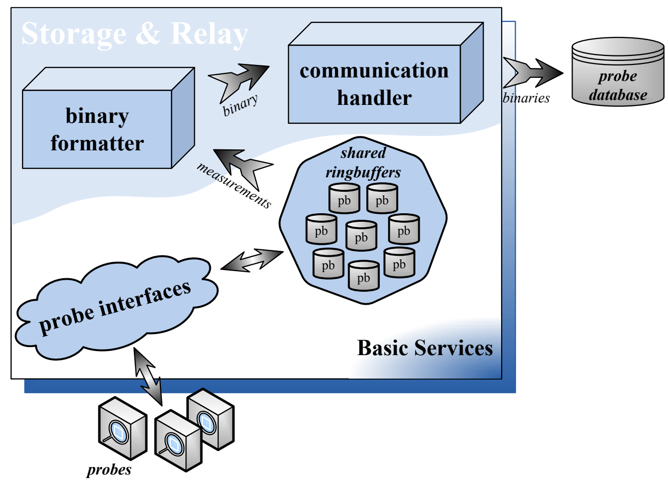


Fig. 4. Structure of basic services

Together these parts take care of all the data management of the tracing as described in Figure 2.

The monitoring services offers a set of readily provided interfaces and probes to attach to the basic services. The actual services at this layer are used to capture and monitor different values, such as memory consumption and CPU usage, and their evolution in the system. Many of these basic monitoring services are provided as ready probe components in the implementation of the PF, including CPU load, memory consumption and network traffic monitoring. Further, they provide simple interfaces for building new monitoring services on top of them without the need to concern with the complex internal details of the data management.

The top layer, test services, is the most implementation dependent and is where the system specific functionality can be build. For example, functionality can be built to inject test data into the system, use a provided set of monitors to see how the system behaves and store the test results using the basic services. Similarly, in a running system the same monitors could be used from a test service (or more accurately, built-in functionality) that adapts the system's runtime behaviour and use of components based on thresholds set for monitored values such as memory consumption, CPU load and network traffic patterns.

An important concept for this type of analysis is that of testability. Although generally associated with testing, it is also relevant in many ways to any requirement to capture information about system execution for analysis. Testability is commonly divided to two main properties: controllability and observability [20]. Controllability is the ability to control the system execution, for example, in order to make it take a chosen path. Observability is the ability to capture information about the different properties of the system (such as events and messages). Although a monitoring framework such as PF is mainly concerned with observability, also controllability is important in order to build support for dynamic and effective tracing as well as services that make use of this trace functionality. For example, in order to insert "proxy" style probes to capture communication messages between system components, the design must support the reconfiguration required, possibly even during runtime. Solutions to address this have been discussed in more detail in [2].

IV. Implementation

A prototype of the PF has been implemented for embedded real-time Linux systems. This platform was chosen as it provides an interesting and realistic platform for the implementation of this type of software, with both possible issues and available options. These issues include the strict timing requirements and limited resources inherent to the embedded real-time systems. Yet, even as we are dealing with embedded software where we know all the running software beforehand it needs to be possible to access the whole platform including the kernel. With Linux as the operating system, this is particularly easy as the whole operating system is open source software.

A. Setting for the implementation

The setting for the implementation is considered to contain the following elements: run-time monitoring, Linux environment and embedded real-time environment. The setting for the framework entails various interesting challenges. The next sections depict some of those challenges.

1) Run-time monitoring: The goal of run-time monitoring creates challenges because is makes the execution somewhat unpredictable. Generally, the system has limited resources and many consumers "competing" for them. As such, any kind of extra activity in the system can have severe consequences. The extra activity refers to the act of monitoring, which in a software-focused case entails some code being run in order to capture data. The monitoring can be understood as data collection, and instrumentation is a way to realize it through software. The monitoring process in whole changes the targeted system, software monitoring always consumes resources and inflicts an overhead to the underlying system. Depending on the context this might pose a problem.

The run-time monitoring states that monitoring, tests, data collection, etc., are conducted while the target system performs its appointed functionality. In other words, everything happens while the target system is executing normally. However, the concept of "normally" can be bent a little as certain liberties do affect cases where some specific property is being monitored. For those cases, it is only viable to focus on the necessary items while the rest can be ignored. In practice: a specific part of the target, in this case it could be a sub-program of some sort, is isolated and only it is run in the system, while the rest is substituted with "stubs". Still, the aim of run-time monitoring is to stay true to the actual real-life end deployment, and limit the overhead of monitoring, or at least control how the overhead occurs in the target. Because of this, the control aspect of testability is important. For a highly controllable system, the choices of instrumentation leave room to better manage the overhead in the context of the services built to make use of the provided tracing possibilities, as well as to configure more dynamically the used trace services.

The importance of run-time monitoring is diverse; the implementation is seen in its natural environment and several aspects of it can be measured and analyzed. Testing can be done against the "real thing", therefore, unexpected issues have a possibility to surface. Wegener et al. [21] mention a few important issues that can be obtained through run-time monitoring. These are the dynamic aspects such as the duration of computations, the actual memory usages of the program during execution, and the synchronization of parallel processes. These aspects are of special interest for real-time systems as resource allocations might be hard to determine beforehand, and estimates for resource consumptions are only suggestive, and therefore need to be verified. As such, run-time monitoring also provides support for a wider part of the system lifecycle, including in-field diagnosis, where the only option available is to use the available run-time services or otherwise lose the information of interest (such as failure state and its cause in deployed systems). Where information is needed for a run-time adaption of system behaviour, run-time monitoring is again the only possible solution.

2) Linux environment: The Linux environment indicates that the system has either a limited (customized & embedded) or a full Linux operating system (OS), with support for network access, filesystem, scheduling, etc.

The Linux OS is open source, and so is the Linux kernel, with several distributions readily available depending on what flavour one likes. The distributions aside, the core of Linux is the monolithic kernel. This core is quite versatile and it is autonomous from the rest of the OS. The monolithic means that the entire kernel is run in a privileged kernel space in a hypervisor mode, as opposed to user space that has relatively restricted execution possibilities. In a nutshell the difference is that a program in supervisor (kernel) mode is trusted never to fail, and this is not questioned or accumulated for, where as in the user space the failure is an option and there are measures for managing those failures without a total system crash. Because of this, a general guideline for when operating in Linux is to perform actions in the user space if possible, as kernel space is strictly reserved for running the kernel, kernel extensions, and some device drivers [22].

a) Modularity: The modular nature of the Linux kernel is worth mentioning, because it is possible to dynamically load and unload executable modules to the kernel at run-time. This modularity of the kernel is at the binary image level of the kernel. What this means is that after the code comprising the Linux kernel is loaded from a binary image at boot up, it is possible to dynamically load modules to expand the functionality of the current image at run-time – as opposed to rebooting with a different kernel image. The modules allow easy extension of the kernel's capabilities as required. However, dynamically loadable modules incur a small overhead compared to building the module directly into the kernel image. On the other hand, dynamic loading of modules helps to keep the amount of code running in the kernel space to a minimum, useful for example to minimize the kernel's memory footprint for embedded devices with limited hardware resources. Modularity can be used to address the run-time inclusion of PF services, when there is a need to consider general resource constraints and there is need to disable external trace components used with PF for general operation, while maintaining the option to enable these during runtime.

b) Linux kernel: On the implementation level of the Linux kernel the used programming language is C, and as a big part of the kernel deals with device drivers and other hardware-oriented functions, the C language is a fitting choice for all this hardware oriented programming. Due to various constraints, it is generally considered that any code expected to run in kernel space has to be C code. Additionally, the different programs running in the kernel are often separate processes, and user space contains various different processes, and programs implemented with numerous programming languages and environments.

As mentioned the core of a Linux system, kernel, is not dependent on the rest of the OS, therefore it remains relatively the same between different hardware platforms. Also, what is handy about the kernel is that most of the functionality is highly abstracted. Because of these, the gap between host-target development common in embedded systems is not so wide in Linux environment. This gives the option to implement generic services to support capturing information for any system and application running on top of a Linux based system. Thus, some services for OS level information capture can be provided as generic, while providing support for application specific trace implementation.

3) Embedded real-time environment: Another factor in the environment is the embedded real-time aspect. Embedded real-time systems possess unique attributes, and requirements. These two attributes, embeddedness and real-timeness, contribute to the complexity of the environment.

The real-time aspect of the system dictates that there are constraints placed on the timing of the actions the system performs. This can be defined as having performance deadlines on computations and actions[23]. Knowing the state of the system has conventionally been a great help in testing the system and in verifying its behaviour. The system being real-time might cloud this knowledge, the knowledge of the internal functionality of the system for a given point in time, as the possible variations in the system's execution paths quickly approach infinity when time is taken into consideration.

The other side of the equation, embeddedness, refers to systems that interact with the real physical world, controlling some specific hardware, such as a cell phone [24].

In combination, embedded real-time systems are systems built with a specific purpose in mind. These systems specialize in fulfilling that purpose, often with the minimal possible cost. The environment where these systems operate is dynamic; computational loads are unpredictable yet responses have to be provided according to precise timing constraints [25]. The minimization of the cost often results in scarce operating resources and having just enough resources is quite common in embedded systems. The resource constraints common to embedded systems are due to limited physical space, weight and energy usage, and cost constraints. The strict constraints associated to embedded real-time, along with the environments these systems operate in, offer challenging design and implementation choices. Schulz et al. [26] list characteristics of the current generation of these systems: continuing increase in system complexity, diminishing design cycles, tightly integrated mixed hardware and software components, and the growing use of reconfigurable devices.

The features of the embedded real-time environment that are considered important in the context of the Probe Framework are resource constraints, scheduling frequencies and adaptability to data extraction, timing alterations and relaying possibilities inside the target system.

4) Restrictions forced by the environments: The issues common to embedded systems and real-time systems, such as low memory, limited CPU power, timing requirement, etc., are valid restrictions for the implementation. Also, the Linux environment and the run-time monitoring have their share of challenges for the implementation.

However, the biggest challenge is most likely the indeterminism brought forth by the nature of environment. The reason for this is that the execution has dynamic elements that can be estimated to some extent but not to the extent to be irrelevant. Another factor is that each embedded real-time system is unique in context, and preparing for a wide variety of embedded systems is challenging. Because of this, several generalizations and abstractions are needed for adaptability to be able to cover every system. Deciding on the generalization and abstractions, i.e., the common factors, for embedded real-time systems need a set of restriction to be meaningful. Here the Linux environment offers the needed focus point for that purpose. The possibilities offered by the Linux kernel and OS outline a set of functionality for managing the general required operations for system instrumentation. This set of functionality is the basis for our prototype implementation Probe Framework.

A big restriction concerning the run-time monitoring is the issue of a potentially huge amount of data, as the system can execute at the speeds of several megaticks per second, and keep running for a long time, it is clear that the amount of collected information can accumulate fast. When combined with the low amount of memory and limited disk space of typical embedded systems, along with the timing requirements of real-time systems, the combination requires special attention. Another restriction is the used programming language. Considering the C dependency of the kernel, it is easy to see why most of the software in Linux is done either in C or C++. Through this factor, it is natural that the choice for implementation language of the Probe Framework prototype is C, and C++ for parts where it is fitting. Also, the partition of the user/kernel space in Linux causes the need to manage each side's monitoring needs separately and as conveniently as possible.

B. *Main parts of the implementation*

The focus of the prototype implementation is on the basic services layer, because the basic services are general for all the probes, and offer the support for fast implementation of the upper level services. As mentioned, the basic services comprises of three parts: the probe interfaces, the binary formatter and the communication handler, see Figure 4. Here the implementation of these three parts is covered, starting with the probe interfaces.

Although conceptually one, the actual implementation of the probe interfaces in basic service layer is divided in two. The major reason for this is the way execution in operating systems typically takes place, in either user space or kernel space, refer to Section IV-A2. This separation also acts as a divisor for the probe types, resulting in a split between kernel probes and user probes. Additionally, in Linux as well as most modern OS's each user space process runs in its own virtual memory space, and thus cannot normally access the memory of other processes nor can other processes access its memory [22]. However, for effective implementation of the PF, all the trace data for a single system needs to be centrally managed. This requires that there needs to be a single component that takes care of the data management for all the probes deployed, either in kernel or user space. This division of implementations, probes, probe interfaces and storage and relay module, is described in Figure 5.

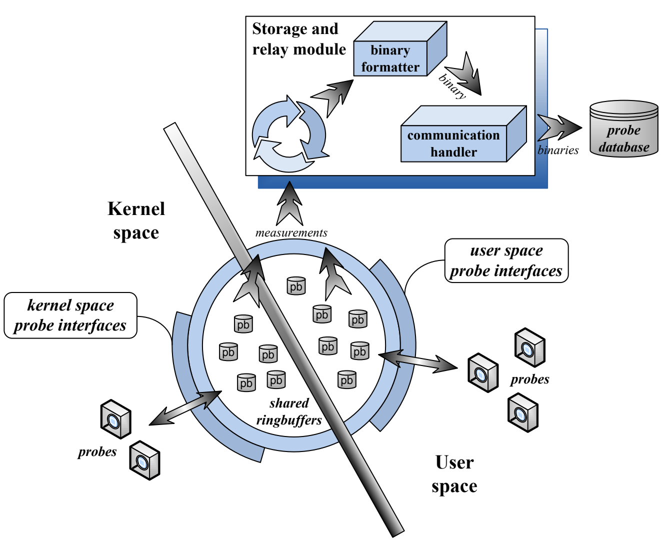


Fig. 5. Division of the implementation

The storage and relay module resides in the user space, conforming to a general guideline for operating systems [22]; perform actions in the user space if possible, as kernel space should be reserved for parts that absolutely must be there as they require special privileges. Kernel code can also crash the whole system with its privileges and thus these parts need to be absolutely secure and reliable. Since we do not need to perform actions with special privileges it makes sense to locate most of the code in the user space. This is also one of the main reasons for why the shared memory regions are used between kernel space and user space, and also inside user space, see Section IV-C5 for details on the shared memory utilization. This was all in order to separate the trace handling functionality, see Section IV-C4 for trace handling information, from the probes and to centralize the trace collected by the probes. This enables the storage and relay module to access the trace, format it and provide it for higher layer functionality or simply relay it to the end storage as requested. All this reduces the interference induced to the target by the monitoring activity conducted by the probe as all the ”extra” processing can be done separate to the probe in its own process. Another benefit for having the storage and relay module, i.e., the basic services, in its own process in user space is that it enables easier configuration of the provided services.

The binary formatter part of the said module is the simplest part of the component; it is as the name suggests a formatter used in changing the collected data to a more manageable form. The reason for the use of binary format is to provide an effective, single format to share the data between different tools, layers and databases, refer to Section IV-C6 for detail on the binary format. The intent is to support probes created in different programming languages, running on different platforms and with strict constraints on memory and real-time requirements typical to embedded systems. Implementing this effectively is not trivial; however, the user is completely shielded from the details by the provided abstraction interfaces. The communication handler is the second part of the storage and relay module. This part handles the data transfer to end storage locations, takes care of the configuration of the storage and relay module, see Section IV-C1 for the configuration, and manages the data extraction from the probe buffers.

In practice the basic services are implemented as a shared library component, meaning that the implementation code needs only a single instance (code segment) to reside in the memory during runtime. This makes synchronizing all the trace data for a system overall much simpler due to only having a single instance of basic services for a system at any time. The library is implemented as a dynamically linked library, which is linked to the components during execution, meaning it is shared also between different processes in the user space. For the kernel space there is a similar component.

The following sections shed more light on the internal functionality of the Probe Framework.

C. Properties of the implementation

The most important properties of the Probe Framework are covered here. The included sections deal with the Probe Framework's configuration possibilities, instrumentation possibilities and the way the data collected by the probes is managed in the framework. Also, the message forming and the used binary protocol is covered here.

1) Configuration of the implementation: In order to cope with a variety of different devices, the configuration possibilities of the Probe Framework are substantial. Each probe can be configured separately, as can be the storage and relay module. Various possibilities for accumulating for the different capabilities of the target system are offered by the Probe Framework. The output possibilities and replacement strategies, etc., used by the storage and relay module are all configurable to suit the system's capabilities.

The major control features of the Probe Framework reside inside a configuration file that is read during the activation of the storage and relay module. This allows configuring the basic parameters such as overall buffers, general policies and storage mechanisms externally. Another layer of control is embedded in the creation of the probes, during the implementation of a probe the creator can use custom settings to define probe specific values or leave them out in which case the default generic values will be used. The probe specific attributes include buffer size, preferred storage location, priority, timing accuracy and presumed output type. Additionally, the creation of output types used by the probe introduces control as it is possible to use prioritized data types for increasing the probability that the collected trace reaches its storage location. In order to address restrictions such as keeping the monitoring overhead low, several policy parameters can be defined. One is the possibility of discarding parts of the collected trace if the basic services cannot run fast enough to relay it to a storage destination. This is further influenced by the priority set to the trace through the configuration. More advanced policies can be implemented inside custom probes, such as sampling or time-interval captures.

The most important part in configuration is when, how and on what condition the data is collected by the probe. This is left open on purpose so that the implementor can choose the most suitable collecting method. Choices need to be made about the frequency of the data collection and preconditioning of the collection events. As an example, the collection could be sampling-based so that every Nth data value is collected. Similarly, some exceptions, values larger or differing from the expected, could work as a trigger for the collection. The choice of how to collect the data and the implementation of this selection is hence forth referred to as the front end of the probe.

a) Configuration file: The main features of our PF implementation configuration are controlled by a configuration file. It allows one to define the storage locations for the collected data and the relative importances between the storage options when multiple options are activated.

The available storage types are memory, file, folder, TCP and UDP. Depending on the type, additional parameters are given to define, for instance, the ip address of a trace server for the TCP and UDP options, or the path for the storage file. The trace server for TCP and UDP in this context refer to (an external) server component that stores the provided data into the probe database mentioned earlier. Continuing on the main types, two of them are quite similar mass storage types, namely the file and folder. The difference between these two are the used replacement strategies; more on these strategies in next paragraph. Another important part of the configuration is the capability to define if a given location is dynamically or statically allocated. There are two options for each storage type. First is the relative priority of each instance compared to the other storage type instances. The second is the allocation strategy, which can be either static or dynamic. The static allocation implies that resources for utilizing that particular location are reserved immediately when the storage and relay module is instantiated. For example, the memory type might specify that 2 MB of main memory can be utilized for storage, which would be immediately allocated and made writeable. In the case of dynamic allocation, the memory would be allocated as required as more data is written.

In the case of the Linux implementation, a notable property is that the Linux filesystem does not discriminate between mass storage devices, or physical location for that matter. Because of this, it is even possible to use a nonlocal file or folder as a storage destination, given that the location is mounted to the local filesystem. This way, it is possible to use external hard drives, networked hard drives, usb-sticks, mmc cards and even memory-mapped files for storage locations, as long as it is possible to refer to it trough a path in the main configuration file. This offers a lot more versatility to configuration.

b) Replacement strategies: The replacement strategy for the used storage is defined in the configuartion file. Depending on the type of the storage, there are several strategies available. For the TCP and UDP types, the strategies are not used because they typically refer to an external trace server and controlling an external entity is not in the scope of a PF functionality embedded inside a system implementation. Buffering strategies would be meaningful but they are different from replacement strategies and we have currently focused on strategies meaningful to embedded functionality. The strategies are therefor relevant to the memory, file and folder types.

For the memory type, the replacement options are *stop*, *restart*, *priority* and *wrapping*. *Stop* means that when the amount of memory allocated has been filled with data, the memory is no longer available as a storage location. With *restart*, when the allocated memory is full, all the previous data is discarded to make room for the new trace. Using *priority*, the data with the lowest priority are removed first to make room when the allocated memory is full. Each data element is stored with a user defined priority as will be described in following sections. The last case, *wrapping*, treats the storage as a circular buffer with varying slot sizes. The next suitable slot is chosen from the buffer for writing.

For the file and folder types, the basic replacement options are *stop* and restart. Folder has additionally *singleflip* and *multipleflipN* (where $N \in \{\mathbb{Z}^+ \cap \{1,2\}\}$). These two new options are variations of the restart replacement strategy. Here N denotes the number of used files. These flips utilize N number of files to divide the allocated space in N parts. The result obtained is the maximum size of a single file inside the folder. When the current file where the trace is stored to reaches this limit, another file is selected and used as storage for following data. After the total allocated space runs out, the first file is removed and a new empty one takes its place similar to the restart case. The benefit of the flips is that it is possible to save more of the already traced data. For instance, if a simple restart is used, all previous trace is lost when space runs out; but with flips, the trace persists longer in the storage. Therefore, when the monitoring ends, provided that enough trace was collected, the folder will contain N-1 files and have at least

$$\frac{(N-1) * total\ allocated\ space}{N} \quad ,(where\ N \geq 2)$$

bytes of traced data stored.

Lastly before moving on, a short note about utilizing several instances of the storage and relay module. The Probe Framework's architecture places no limitations on this. It is possible to have several of them, each with a specific configuration. This way, the probes can choose the storage and relay module instance to use; the id of the major shared memory region the probe registered to during creation dictates the division. This results in a higher level configuration need that is no longer on the scope of the implemented framework. For the interested, Linux offers several ways for achieving this type of control. One method is called resource groups [27]. It could be used in encapsulating and running the storage and relay modules. This way the modules could be differentiated and priority, resource usage, etc., could be set for each allowing for a more managed solution.

2) Instrumentation possibilities: As described earlier, instrumentation is divided into two main types of probes: kernel and user space probes. A distinction can also be made between internal and external instrumentation. Internal refers to embedding the instrumentation code to the software object that is part of the monitored system. In this case the probe is an integral part of the program code. External instrumentation refers to the probes where no modifications are made to the system software itself. Instead a stand-alone process handles the monitoring from outside of the target software. The PF provides support for all these different types of instrumentation. Custom kernel and user space probes and built-in functionality can be created using the services provided at the different layers of the PF. A set of external instrumentation components are provided as kernel probes and processes to collect and analyse generic properties such as task-switches, CPU load and network traffic. More such custom components can also be easily created. All these instrumentation possibilities share the set of basic services that remain unchanged between different implementation possibilities. Therefore, it is simple to analyse the collected trace data, build additional functionality or make other use of the instrumentation data from all different probes and monitoring tools through the provided interfaces.

In order not to limit the instrumentation possibilities, the choice of what to collect, how to collect it and a part of the implementation of this functionality is left open by the Probe Framework. What the Probe Framework provides for implementing the instrumentation, are the basic services that remain unchanged between different implementation possibilities. Therefore, after the choice of what to collect from the SUT and how or if to use sampling or conditioning on the collection, the Probe Framework's basic services are used by passing the trace to it. In short, what is needed is a simple front end, and then the framework's services can handle the rest. Of course, this is overly simplified of what happens on the detailed level, such as calling the probe interfaces and setting the configuration parameters, etc., Still, it gives the bigger picture of creating the probes.

a) Instrumentation types: The types of probes can be divided to kernel- and user-space probes. These include the possibility to use third-party frameworks to capture the trace and link these to the basic services in order to provide a uniform trace over different parts of the system. One such framework for kernel space is SystemTap [14]. SystemTap enables one to add probe code in nearly any location of the Linux kernel, without the need to modify the kernel source code. This is done by code injection during runtime execution, meaning that it can be done while the kernel is running. Use of SystemTap is discussed in more detail in Section IV-C3.

A second possibility is to modify the system source code by adding the needed instrumentation code directly to specific locations. However, the downside is that this method requires that the software is recompiled.

For external monitoring in the case of the kernel is is possible to create a kernel module dedicated solely to monitoring some activity. In this case, as well as the other cases, the basic services of Probe Framework provide the means of passing the collected data to the end deployment location.

These different instrumentation options are supported for both kernel-space and user-space as the services provided by the PF can be linked to any monitoring code or service.

b) Categorizing the probes: Section III showed the layered structure of the Probe Framework. Here the two upmost layers, the monitoring services and test services, are of interest.

The monitoring and test service layers contain what can be understood as probes. These probes are realizations of certain types of instrumentations. Depending on the implementation, probe front end, etc., they are either context dependent or more general ones. The monitoring services contain the more general probes and focus on instrumentations that collect general information from the target system. These services are reusable, and include for instance the possibility to measure the memory usage of different programs or the CPU load generated by the programs. The complexity of these services is quite simple as they usually contain just the collection of some data values. Hence the name, general monitoring services.

The higher level test services are more context dependent, and their functionality is more complex compared to monitoring services. An example of a test service might be, for instance, the collection or generation of inputs to some part of the target, along with the collection of the resulting outputs. Some processing could also take place in the service before passing the results to the basic services for end storage. The implementations are not restricted to just testing related functionality. The service could, for instance, provide useful functionality for adjusting some quality parameters in the deployed system. Although our implementation is focused on the basic services, for different domains, more generic higher level components could also be provided.

Finally, as was presented in Figure 3, it is possible to construct probes that utilize all of the layers. This means that it is also possible, for instance, to use previously made probes to create a new more complex one. This case is mostly relevant for the test services layer, where it is possible to use the services from all the layers beneath it.

3) Linking with external tools: As an example of linking the PF with external monitoring tools, this section describes how we have linked it with SystemTap. SystemTap is a framework characterized to simplify the gathering of information about the running Linux system. It is also said to eliminate the need for the developer to go through the tedious and disruptive instrument, recompile, install, and reboot sequence that may be otherwise required to collect data.[14]

With SystemTap, it is possible to monitor the variables used by the kernel, even altering the variables and performing extra operations inside the placed probes is possible. For controlling this functionality, SystemTap provides a simple command line interface and a scripting language for writing instrumentations. SystemTap can be utilized in most Linux installations as the required components are commonly included in the distributions. For other system, similar framework, such as DTrace, exist as described before.

How the SystemTap works and how the Probe Framework works in conjunction with the SystemTap is depicted in Figure 6. The figure is adapted from [28], and has been modified to include the addition of the Probe Framework.

4) Trace handling in the implementation: This section describes the path the trace data takes through the Probe Framework. The trace data is first stored in an internal ringbuffer inside the PF. To assure correct functionality of the system, the different operations are prioritized so that write operations have a higher priority. This is due to the writes often taking place in kernel-space and reads always taking place in user-space. Since kernel-space is more critical, this is given a higher priority.

After storage in this internal buffer, the trace becomes the responsibility of the storage and relay module. This module runs inside user-space and manages the end storage locations for the trace data. It accesses the shared memory regions and uses a table to access the ringbuffer data values in correct

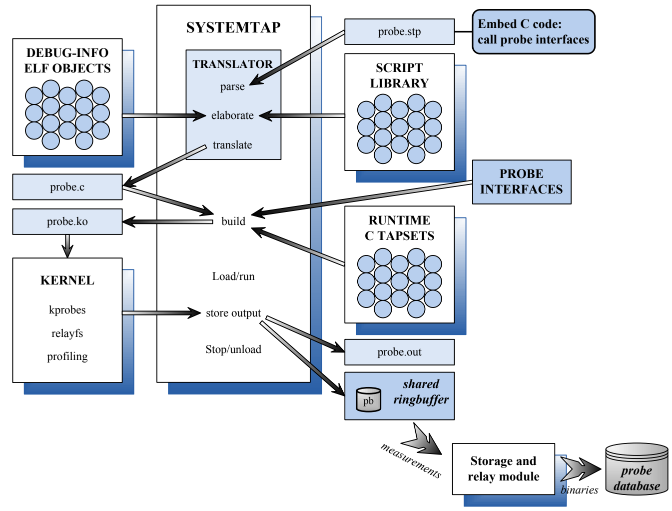


Fig. 6. Probe Framework to SystemTap conjunction.

order. Each probe instance has a specific identifier used to map the correct storage and relay module to the correct probe data, allowing dynamic creation and destruction of probe elements during system runtime. The storage and relay module formats the data into correct format and passes it to the registered storage mechanisms in the order of preference. Failure handling is also included in order to manage cases where a chosen storage becomes unavailable.

For handling special messages including those that denote the start and end information of the provided trace data, special regions in the shared memory are used. These are given higher priority than other parts, as without providing this information first it is not possible to reconstruct the trace data at the end storage location.

5) Shared memory utilization: Due to the large volumes of data that can be produced by tracing specific properties of a system, such as the process scheduler, the performance of the trace data relay mechanism is important. In practice, this requires using shared memory regions between kernel- and user-space. There are basically two fast enough ways to address this requirement. One is that the processes can request the kernel to map a part of another process's memory space to its own, and the other is that a process can request a shared memory region with another process. These shared memory regions are also useable between kernel space and user space processes. The choice made when developing the Probe Framework was in favour of the latter option as it works both in kernel space and user space.

The Portable Operating System Interface (POSIX) is a name used to refer to a group of standards specified by the IEEE. The POSIX shared memory falls under the real-time extensions of the POSIX Kernel APIs [29].

In the Probe Framework, the POSIX shared memory is used for inter-process communication (IPC) between the probes in the user space and the storage and relay module. The choice for using the POSIX shared memory is because it provides a standardized API for using the shared memory, and the implementation is by a set of common C libraries. The API in question makes it possible to use the shared memory regions with reduced effort. The fact that the memory can be easily created, attached to, detached and removed by separate programs in the user space makes it, not only a fast, but also a versatile way of relaying the trace between the probes and the storage and relay module. The shared memory objects can even persist after the creator, the probe of the Probe Framework, no longer exists in the system, thus facilitating versatile collection possibilities for the storage and relay module.

The usage of these shared memory objects in Probe Framework works in the following way: the user space probe calls the probe interfaces of basic services layer to create the internal ringbuffer for itself. This buffer is implemented as a POSIX shared memory object. During the creation operation, the identifier used to refer to the shared memory region, the ringbuffer, is written to a master shared memory region. The identifier of the master shared memory region is known by all parties in the IPC, as such the storage and relay module can then find the ringbuffers and attach to them. This allows the dynamic addition of the probes to the system.

However, the POSIX shared memory is only for sharing memory regions between programs in the user space of the Linux. As such, the RTAI extension was required to obtain a similar solution for sharing memory between the kernel and user spaces.

The Real-Time Application Interface (RTAI) is an extension to the Linux kernel and a collection of a variety of services for real-time programming [30]. In the Probe Framework a specific part of the RTAI is utilized, namely the RTAI shared memory module. This module provides nearly identical functions to those of the POSIX shared memory available in the user space. The difference is that now the shared memory regions are between the kernel probes and the user space storage and relay module, instead of just two user space entities.

Due to the similarity of the RTAI shared memory and the POSIX shared memory, it is possible to use the same memory management method as was used in the user space side. Similar to the POSIX shared memory, a master shared memory region is used in conveying the identifiers of the kernel space ringbuffers to the user space storage and relay module.

6) Trace communication protocol: The Probe Framework uses a predefined set of messages for encapsulating the gathered trace. As such, each single data value is associated with a testcase, data type and other relevant information such as time and order stamps. This is important as the collected data needs to be definable and identifiable for it to offer any real value, by value it is meant that the data can be used for instance in detecting anomalous behaviour and in telling when and where it occurred. The used encapsulation enables the data to be related to a specific context and thus gives it meaning. It is also possible to define various properties of the trace, such as the granularity of the timestamps used for stored values. For optimization purposes, it is also possible to leave out all the added properties (such as timestamps) to focus only on the data where high performance is needed.

In order to support different types of analysis scenarios for the captured trace data, PF provides possibilities to defined customized data types. In its basic format, the data can be defined as either input- or output-data for the system. Further from this the data can be described as either a primitive value (such as a number or a boolean) or a text string. This allows for more advanced analysis of the data that is exported. This type information is only given at the beginning of the trace, and different formats are later supported to compress the trace data where the type is the same for a large number of elements. This again supports high-performace tracing.

V. Experiments and Experiences

To evaluate the PF we carried out two case studies. Both of these are in the domain of monitoring embedded software-intensive systems. We focused on using the monitoring services layer of the PF, and indirectly the basic services through the monitoring layer. We start with describing each experiment and the PF overhead cost on the analysed system. We show what we discovered from the collected trace and how we found the trace could prove useful in a larger sense. In the end we describe our experiences in using PF as a platform for implementing overall instrumentation for system monitoring.

The two case studies we performed were monitoring kernel task switching and the memory usage of different processes. The cases are divided by the type of instrumentation, see Section IV-C2 for the types. The task switching case was done via an internal kernel space probe and the memory usage case via an external user space probe. The memory usage case was conducted on an embedded system that was provided by Espotel[1]. This platform, called Jive[2], is a battery-powered, touch screen equipped PDA type of a device with broad connection interfaces. For the task switching case a typical desktop PC was used.

A. Task switching case study

The task switching case study focused on the scheduling of processes (tasks). In a typical modern OS there are numerous processes running at the same time [22], and the scheduler handles the execution of tasks by dividing the CPU resources to slots and distributing these slots to the tasks. Our goal was to build a monitoring probe to capture the information on how the task switching is performed with the given usage scenarios. The visibility of the scheduling activity in this scenario is strictly for the kernel space only, and as such, the monitoring had to be implemented as a kernel probe. For this case study, we collected three types of events:

- Task activate
- Schedule
- Task deactivate

[1] http://www.espotel.fi
[2] http://www.espotel.fi/ratkaisut_jive.htm

The schedule event means that the running task is switched to another, the meanings of activate and deactivate are a bit more complex. The activate and deactivate denote that the task is moved to or away from the run queue. For simplicity it can be tought that these two events tell when the task can be run, i.e., scheduled.

1) Internal, kernel space probe: In order to keep the case simple, the processor was set to utilize only one core during the instrumentation. A multi-core task switching instrumentation is possible, but it would overly complicate the case as it requires that additional data is collected to identify the core.

The instrumentation used in this case is an inline probe implemented via SystemTap. The code that uses the framework's probe interfaces is added to the SystemTap probe script as embedded C. For details, refer to Section IV-C3. The targets for the instrumentation are the three events, therefore, the first step is to match them directly to the kernel code. The matches for the task activate, schedule and task deactivate for kernel 2.6.24 are found from /kernel/sched.c under the kernel source tree, as show in Table I.

TABLE I
MATCHES FOR THE SCHEDULING EVENTS

Event	Line no.	Function
Task activate	1004	static void activate_task()
Schedule	3636	asmlinkage void __sched schedule()
Task deactivate	1016	static void deactivate_task()

With the locations of the matches known, the SystemTap is instructed to collect the desired data and insert the code that invokes the kernel probe interfaces of PF to these locations. Also the creation of the probe instance, initialization, etc., are placed to the appropriate locations in the SystemTap probe script.

In each of the locations targeted by the probe, the collected data are the involved task ids, states of the tasks and the current CPU cycle count. Along with those, the time and order stamps issued by the Probe Framework are also stored.

As the instrumentation is done using external instrumentation it also serves to provide a generic reusable kernel monitoring probe for future use when task switching needs to be analysed. Figure 7 illustrates the instrumentation conducted in this case.

2) Intrusiveness of the instrumentation: In this case, as task scheduling happens numerous times each second, the used instrumentation is extremely intrusive. There are bound to be consequences due to the instrumentation code. As we want an accurate picture of the task scheduling, all the events need to be collected and no sampling can be used. Therefore, the overhead is so high that the probe is only useable temporarily for purposes such as diagnostics or to provide basis for performance analysis. In this case we do not have any hard real-time requirements so the temporary inclusion of the probe and the temporal effect it poses on the execution is acceptable. Due to the use of SystemTap this probe can also be enabled (included) temporarily in a running system and disabled (removed) when the required diagnostics

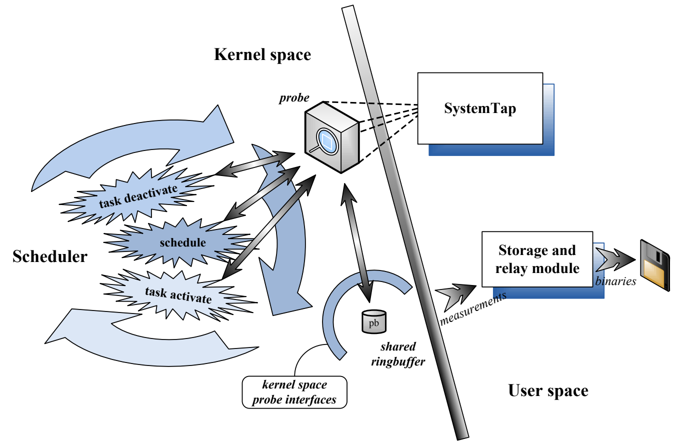


Fig. 7. Task switch instrumentation.

data is collected. Thus, it shows how it is possible to create PF probes that can still be included in the system probe set also in deployed systems while they are only used in ad-hoc style during system execution. Concerning the analysis results, it needs to be taken into consideration that the probe code will consume part of the CPU time, causing skew in the time interval trace, and that the storage and relay module that runs in the user space as a normal task will appear on the obtained task switch trace.

a) Processing overhead: The overhead of the PF was measured by capturing system timestamps as jiffies, these jiffies describe system time/clock ticks as 4ms intervals. The jiffies were obtained from the /proc/stat pseudo-file, see Section V-B for more information on the pseudo filesystem. The stamps were collected at the beginning of the instrumentation and at its end. To obtain a reference point, the duration of the instrumentation was measured, and then the same captures were done in a system without the instrumentation, using the measured duration as a time interval between the captures. To give a better picture of the effect the instrumentation had, the overhead is given as the reduction caused to the true idle time of the target system. The overhead is calculated with the following formula:

$$\frac{\left[\frac{DI}{J} - (CJE - CJB)\right] - \left[\frac{DI}{J} - (CJEI - CJBI)\right]}{\frac{DI}{J} - (CJE - CJB)} * 100\%$$

DI = Duration of the instrumentation,
J = Duration of a jiffy,
$CJEI$ = Captured jiffies at the end of instrumentation,
$CJBI$ = Captured jiffies at the beginning of instrumentation,
CJE = Captured jiffies at the end of idle,
CJB = Captured jiffies at the beginning of idle.

Figure 8 clarifies the used formula, note however that the overhead does not accumulate linearly as the figure might suggest. What matters here is the total accumulated overhead, not the momentary overhead values, even though those values might pose an interesting topic for further study.

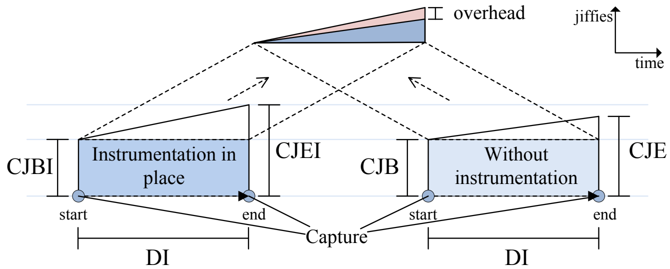


Fig. 8. Processing overhead

The calculated overhead for this case was 16%. Overall, this could be considered a high cost for instrumentation. However, for an analysis case where the monitoring instrumentation is very intrusive, i.e., in one of the most frequently executed function of the kernel, we do not consider this to be a bad result at all. This probe is only intended to be used as a temporary analysis aid and not as a fully included production class feature.

b) Experiences: The Probe Framework manages to obtain the targeted attributes. The three events are captured, and the associated cycle counts, task ids and states are collected. The internal probe is successfully injected to the running kernel via SystemTap. The probe is placed into the correct locations in the running kernel, and correct values are extracted. On the user space side, the storage and relay module manages to handle the data collected by the probe, but not without some initial difficulties. We had to optimize our initial code to address the large amount of data captured. However, in the end, the obtained trace is successfully stored according to the configuration on the mass storage.

In our first try, the internal ringbuffer overflowed, causing corruption in the final stored trace data. This was due to the huge amount of data from capturing all task switches in the scheduler. By modifying the monitoring system to use a larger ringbuffer and with the use of the shared memory regions, the experiment was finally carried out successfully and both the trace capture and the storage and relay module were able to perform their duties without error, producing a correct set of trace data.

c) Utilization possibilities for the trace: In our case study, we used the obtained trace for different purposes, including the characterization of the process load running on the system as described in [31], for analysing task blocking and scheduler performance.

The used instrumentation provides a fingerprint of the combination of processes run of the system during instrumentation, a load characteristic. Based on this data the internal dependencies between tasks are revealed, such as task execution order dependencies. A practical case conducted by Jaakola [31] showed how this kind of data can be used in simulating how the load characteristic could be executed on a different amount of processing units. With his method, it is possible to simulate how the number of CPUs affects the time required for running the same task set as was run on the instrumented system. For testing purposes, the obtained trace could be used in detecting if priority inversion is taking place. A lower priority task blocks a higher priority task due to the higher priority task waiting for a resource the lower priority task is utilizing. The trace can also be used in estimating how well the scheduling performs and if the time slots dished out by the scheduler are of the adequate length, i.e., the scheduling itself is not restricting the performance.

B. Memory usage case study

The memory usage instrumentation case is defined more specifically. The focus is on user space instrumentation, and the instrumentation target is the procfs, process information pseudo filesystem, which is an interface to the kernel data structures and provides information about the processes on the system. It is located under path /proc in a typical Linux distribution. Actually, the procfs is not really a file system because it consumes no storage space. It needs only a limited amount of memory, as all of /proc resides in the main memory, not on disk. It offers an easy access to information about the processes on the system. Originally, as the name suggests, procfs was meant for kernel and kernel modules to send information about processes' to user space. Nowadays, it is used by all of kernel to report anything interesting to the user space. As such, it is a popular method for user-level to obtain information on the system internals such as the current memory consumption.

The targets of interest here are two memory usage illustrating attributes:

- Total amount of used physical memory
- Total amount of used (memory) swap space

The swap space in Linux context is no different from any other modern OS; it stands for the chunks of memory, normally referred to as pages, that are temporarily stored on the hard disk to cope with the limited amount of available physical memory. Swapping, the process of utilizing the swap space, works by copying a page to or from the preconfigured space on the hard disk to free up or populate a page on the physical memory. With the combination of the physical memory and the swap space usages it is possible to derive the amount of used virtual memory. The used virtual memory is the sum of the used physical memory and the used swap space. Before proceeding further, it is good mention that pages in swap can have duplicates also in the physical memory, thus making the actual values a bit inaccurate.

1) External, user space probe: Per specified, the probe used in this case is an external, user space probe. It is an executable that contains the probe front end and the code that calls the user space probe interfaces. This executable is run in the user space with all the other tasks.

As the procfs, target of the instrumentation, houses the wanted data, the probe needs to read the file that contains the total amount of used physical memory and the total amount of used swap. The pseudo-files in procfs reside in memory and have no content until they are read.

Unfortunately, in procfs there are no direct equivalents to the wanted values, but there are four attributes in a pseudo-file meminfo that can be used to derive the target attributes.

The four values, targets of interest, inside the /proc/meminfo are:

MemTotal: X kB
MemFree: X kB
⋮
SwapTotal: X kB
SwapFree: X kB
⋮

In order to capture these values the external probe opens the /proc/meminfo file and reads the content; a snapshot of the values taken when the meminfo was opened. After the probe parses the four values and performs simple subtraction operations to obtain the used physical memory and the used swap:

$$MemTotal - MemFree = MemUsed$$
$$SwapTotal - SwapFree = SwapUsed$$

The probe proceeds by storing the $MemUsed$ and $SwapUsed$ values to its ringbuffer. Along with the two attributes, the time and order stamps issued by the Probe Framework are also stored. The capturing of the four values and the processing is nested inside a loop that iterates every three seconds until the probe executable is terminated. The creation of the probe instance, initialization, etc., are also done inside the probe executable.

Figure 9 displays the instrumentation setup used in this case study.

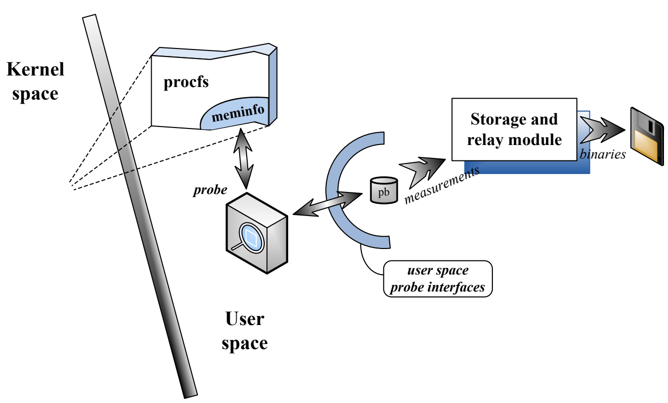


Fig. 9. Memory usage instrumentation.

2) Intrusiveness of the instrumentation: The instrumentation in this case has plenty of leeway, as the intrusiveness can be better controlled. The external probe is a normal task in the system. To minimize its effect, the capture frequency for the target attributes was considered. As reading the meminfo file causes extra operations for the kernel, it needs to populate the pseudo-file with values from the memory; the overhead of the read of meminfo is proportional to the read frequency. Similarly, the parsing the probe performs along with the normal operations of using the framework's probe interfaces causes overhead that is proportional to the capture frequency. Also, the fact that the nature of the wanted information does not specify how often it needs to be captured, as opposed to the previous case where all the events had to be captured, leads a result that time sampling is used to limit the probe's impact on the system. The external probe is set to sleep at least three seconds between captures. The three seconds is an approximation because the processing in the system and the processing of the probe itself induces indeterminism.

Considering the frequency of the capture and the small extra work created by the data collection, the overhead of the probe should remain fairly low. Still, some periodic delays to the execution of the processes running in the system are expected. For this case, the overhead caused by the used instrumentation was measured using the same method as in the previous case. The induced overhead was 2%, which is not overly much and could still be further improved with more efficient integration with kernel functionality.

3) Experiences: The Probe Framework succeeds in obtaining the targeted attributes, total physical memory usage and total swap usage. The external probe processes the /proc/meminfo pseudo-file successfully and calculates the targeted attributes. The storage and relay module manages to handle the data collected by the probe. The obtained trace is also successfully stored according to the configuration on the mass storage. Overall, the framework performs as expected, no errors or exceptions rose, and the targeted values were collected.

a) Discoveries made from the trace: A closer inspection of the obtained trace, after feeding the binary file containing the trace to the probe database and reviewing the actual values of MemUsed, SwapUsed and the accompanying time and order stamps, revealed several issues. First of all, the order stamps indicate that the external probe's internal ringbuffer did not overflow, i.e., all the values captured by the probe made it to the probe database. Second, the time stamps of each target value are all neatly three seconds apart, as specified by the configuration. And Third, an interesting discovery was made by inspecting the SwapUsed values – they were all zero. There are three possible explanations for this: either the external probe failed to process the target values correctly or the storage and relay module misbehaved silently and stored incorrect values or the Linux running in the Jive simply is not configured to utilize swap space. To ensure that the trace was not erroneous, the Jive was inspected, and it turned out that swap space was indeed not in use. The fourth observation was that the MemUsed values in turn all remained under the 64 MB limit, the total amount of physical memory present, and appeared consistent in magnitude.

For effective compression of the trace data values, the storage of two different types of data values at each timestamp proved problematic. The PF implementation only provides options for compressing the data with an expectation that single data elements are present. This works in a case like task switching, where only the id value of the next task is of interest. However, in a case where several values (MemUsed and SwapUsed) are stored at each timestap, this does not work. Instead, more powerful support would be needed for formats where several data elements are stored in always the same order but there are more than one type of element. This was a design choice in the prototype implementation and could be addressed with an update version.

Additionally, the fact that the swap is not in used by the Jive raises question: Would it not be better to only collect the physical memory statistics? Indeed, it would. But the idea here was to create a generic external monitoring probe that can serve as a monitoring service in other devices too, where the swap could be utilized.

b) Utilization possibilities for the trace: Our intent was to use this information for observing memory leaks and growth of consumption over time, similar to audit tests inside a running system as described in [2]. This type of memory problems are considered to be among the most common and difficult to debug due to the long time they take to develop [32]. Thus this type of information is useful to have available to describe the development of the symptoms and to analyse their cause over time.

Worth mentioning is also that the trace can be used for estimating how a new program added to the system or a modification to an existing one changes the memory usage. The trace could help in answering if the available physical memory of the system should be increased to meet the requirement placed by the tasks being run in the system.

C. Experiences in PF development and use

Few of the hardships faced in developing the PF and experiences gained from those are discussed next.

1) Testing and debugging of the framework: A major challenge proved to be the debugging and testing of the implementation itself. As the Probe Framework deals with a large number of error-prone functionality, such as concurrent accesses, shared memory and a great deal of pointer operations in the memory, and contains several internal as well as external interfaces, the testing and verification of it was considered quite important. The testing and debugging had to be done not only in the host environment but also in a system that represents the assumed target environment. The Jive served as an example of such a system.

It took a lot of effort to prune the Probe Framework of errors and to assure its correct behaviour. For testing the tool, C++ unit tests [33] were utilized in testing the user space part, and lots of debugging by hand was carried out. Even with all the effort, the possibility that the tool contains errors still remains. This is the fundamental problem when it comes to testing any complex system. By exaggerating a little it can be said that in every software application there are bound to be lingering errors, because if that was not the case there would be no need for testing to exist in the first place.

As such, this highlights the usefulness of providing a component such as PF, which can be highly intrusive in a system, and needs to work in critical parts such as the kernel-space functionality. Thus re-implementing all the required services in different projects is not feasible but providing a well tested and optimized version would be of great benefit, such as our prototype implementation in the used case studies.

2) Host-target separation: The first hand experience on developing on the Jive platform made the complications associated to host-target separation abundantly clear. The reason was that this separation created large obstacles for the development and trials of the Probe Framework, as the many dependencies in the used compilers, capabilities of the host and target systems, and the used C libraries resulted in many complications.

If there is a lesson to be learned from this, it is to ensure that the system where the development takes place resembles the target system as closely as possible. Of course, this does not mean that the host system should be hard to access and contain the challenging features of embeddedness and real-time. Instead, it means that there ought to be easy connectivity and fast trial possibilities available, either on the actual target or in a very close resembling simulator or emulator solution. This also supports the usefulness of having a readily provided trace component available for use in different environments. Even with the common factor of the Linux kernel, and an overall similar OS, between the host and the target, the development was challenging.

3) Trace handling: Two particularly challenging issues in the development and trials of the tool were both faced in relaying to collected trace. The first on was the crossing the of the kernel/user space boundary. Even though there are a few methods for achieving it, there is not much information available on the matter, due to this and the unfortunate lack of time the finding of the most effective method had to be left for another time. As mentioned, the Probe Framework had to result to an outside solution for communicating over the kernel/user space barrier, several issues contributed to this choice but the most notable one was the lack of time to implement a custom solution for it.

The other hardship tied to the trace relaying was encountered in the task switch instrumentation. The problem encountered was two-fold; the first part was that the overwhelming amount of collected data required special measures to be taken. For Probe Framework this meant optimizations to certain functionality, but it also increased the intrusiveness of the data relaying as more complex functionality had to be implemented. Overall this kind of instrumentation that collects ”too much” trace is not very feasible to begin with, and certainly cannot be left active in the system all the time.

The latter part of the encountered problem was that because of the nature of the task switch instrumentation, some instability was present. The reason for this is that the scheduling activity cannot be allowed to be blocked. In other words, the system will hang if the probe cannot access its internal ringbuffer and associated pointers, due to the storage and relay module holding the synchronization lock. As a relieving measure, the probe had to be set to ignore all used synchronization methods and to behave as if it was the only one operating on its shared ringbuffer. Initially, this caused the instability, as the data pointed to by the read and write pointers did not stay consistent with the synchronization gone. To reduce the instability to unnoticeable, the storage and relay module had to take precautions not to alter any of the data while the probe was operating on it. This was achieved with a manner of a safe zone that was utilized in the storage and relay module to keep the read operation from reaching the write location, and thus interfering with it.

4) Inherent indeterminism: On a more general level, one insight gained is that preparing for the unexpected is not always feasible. For instance, the case might be to perform an instrumentation or testability related activity in a predefined time slot to minimize the effect it has. As this can be extremely hard, rather than trying to hit a specific time instance, it might be more beneficial to aim for a certain event slot in the execution sequence of the target system. The reason for this is the indeterminism present in the system, the performance and usability improving features, such as the buffering of the write operations, the preemption of tasks, etc., change the time instance when the instrumentation's resource consumption takes place. The indeterminism also contributes to the hardship of estimating the consequences of testability functionality because it induces variations to the observations.

For dealing with the indeterminism, it is beneficial to have several possibilities for implementing the instrumentation. Focusing on the software approaches is not the only way. For a more versatile solution, the hardware-based approaches and hybrid approaches are worth considering. However, this creates another trade-off for testability, by multitude of instrumentation choices the factors that need to be considered also multiply. Thus one goal for future work could be to bring also these types of tracing methods into the framework to support providing a single unified trace also in a combined hardware and software tracing domain.

5) A general instrumentation approach: After our trials with the two case studies, we can say that the framework provides a reliable and efficient instrumentation interface with high potential for reuse. We implemented two highly reusable and generic probes. Due to their very generic nature, they can be reused as is or with minor modifications in other contexts. As a generic framework is also bound to be used more frequently and by more people and projects, the code will also become more reliable and optimized over time than separate custom solutions. That is, the more the PF is used the better it becomes. This makes it more likely that found problems are in the system itself and not in the instrumentation code.

The best side of a general, reusable solution is that the more it gets used, the better it becomes. This is true for the Probe Framework, as new probes are created, monitoring services and test services alike, the usability of it increases. For instance, even though the two case studies used in this work were simple, now that the probes have been created they can be reused with very little effort or modification. The ready monitoring services of the Probe Framework also lower the required understanding of the target system. However, a downside related to the usability of the Probe Framework is that the current interfaces of utilizing it might be a bit complex and should be considered for improvement.

As noted earlier, the basic services of the PF have also been implemented on the Java platform. As the PF's implementations both share the same file formats and protocols, we have also been able to successfully use them together. In this sense, through the shared information database storage and export facilities it is possible to get a view of systems with varying component implementations. It is our experience also from these implementations that the simplicity of the provided interfaces is a key to their easy adoption. They must be simple and easy to use and not get in the way of the developer. By hiding all the complexity of trace storage, processing and access behind simple interfaces the PF becomes also more convenient to use. And, that is what the PF aims for, to be a general reuseable approach for instrumentation that lets the developers better focus their efforts on implementing the actual product rather than spend overly much time on creating ad-hoc instrumentation solutions.

Regarding the usability of the Probe Framework, it is not limited to the context of embedded real-time systems. Those attributes are merely something that create a challenge for the PF, i.e., limit the available resource, etc., and in no way limit the environment that the PF concept is suitable to. Similarly, the prototype implemetation being Linux specific doesn't indicate that the PF concept couldn't be used in a different OS. The PF's mentioned Java platform implementation for instance is not limited to the Linux environment.

VI. RELATED WORK

There are no direct equivalents for the Probe Framework but it does have similar aspects with many other approaches for instrumentation. In general, the instrumentation solutions remain as specific and customized as the systems they target, and as the Probe Framework is just that, a framework for building a more meaningful functionality, it is hard to compare it to other specific solutions. The Probe Framework is considered as a support provider, a building block to be used with other tools and custom handmade solutions, not a replacer for the other instrumentation solutions.

One particular instance, quite similar in the used methods to Probe Framework, is introduced by Chodrow et al. [34]. Their tool is for specifying and monitoring the properties of real-time systems during runtime. Their tool uses an "external" monitor that collects data from events. These events are somewhat similar to the probes used in the Probe Framework but not as versatile, as the events only count the occurrence of some "event" inside a given task. The monitor they use is similar in functionality to the storage part of the storage and relay module offered by the Probe Framework, but not much is said about the storage possibilities. Another similarity is the use of shared memory. Their tool uses a file-mapped memory, a shared memory region, to pass data from the events to the monitor. This usage of the shared memory is similar to that in the Probe Framework, but it is much simpler and lacks the timing and other associated data that is stored by the Probe Framework. Also, the methods they use for data collection, inline or external entity, are much like the ones used in the Probe Framework. An interesting point they make is to use a tool such as theirs for other functionality than just testing. They give an example: "One can envision an environment where an application task specifies a user-defined function that is invoked by the run-time monitor in response to an event occurrence."[34]

It can be argued that any tool meant for monitoring, testing, data collection, etc., is similar to the Probe Framework, and yes it is a valid deduction. The whole concept of system instrumentation offers a wide field for different tools, and as the Probe Framework can be customized and used for specific purposes on that field then there are bound to be similar existing tools to that specific instance. Yet it is not meaningful to compare the possible implementations realizable with the framework to existing ones, and the sheer number of possible implementations and existing tools makes it unfeasible in this context.

VII. Conclusion and Future Work

The Probe Framework described in this paper provides the means to build and later on reuse system instrumentation approaches effectively and reliably. It provides support for the basic requirements of storing, relaying and accessing data. More advanced needs such as processing, monitoring and building new functionality to use the traced information are supported by the PF's higher layers. Two cases studies where the PF was used were carried out to validate the different uses of the framework. These cases used the provided building blocks and interfaces to build generic, reusable probes for gathering important system information.

The nature of embedded systems is that there is little consistency between different devices, having led to creation of customized solutions for information access. Here, we have shown that for a system where the PF is available it provides a basis for a uniform instrumentation solution. Generic probes can be reused across systems and new ones implemented by using the provided building blocks and interfaces. The reuse of the framework and probes thus leads to reduction of the implementation effort and also to increased reliability as the found problems are more likely to be in the system itself than the instrumentation code.

For easing the lifespan testing, diagnostics and management of the target system the Probe Framework can be very useful. Given that the Probe Framework is intended to remain in the target system after deployment, it can provide its services during the targets lifespan. Therefore, it can hasten the detection of the possible problems and offer the testing and monitoring services during the targets lifespan. In practice, the Probe Framework requires that the shared library, storage and relay module and the various probes remain in the target, to provide its services after the target has been deployed. Overall, the space requirement of the Probe Framework is minimal, but naturally its presence will affect the rest of the system and should be carefully considered.

As always, there is room for improvement and the list of potential improvements and new features contains several higher-level properties, as well as implementation level optimizations. One interesting high-level improvement is a way to extract internal errors and exceptions from the Probe Framework itself. Voas and Miller [35] mentioned that it is naive to think one can get the monitoring correct when the software under analysis is lacking (no software is seen to be bug-free), as such being able to distinguish between errors of the target and the framework is beneficial. The implementation of this feature could be via predefined "macros", premade messages, for quick and simple reporting on the internal errors and exceptions.

Another desired property for the Probe Framework is the ability to control and configure it remotely. Systems, embedded in particular, are often physically situated in hard-to-access locations. For instance, mobile base stations are scattered around the country, cities, rural towns, etc., and if one happens to malfunction, sending an engineer without any prior knowledge about the problem to do maintenance and repairs will be costly. With remote configuration, the Probe Framework could reside in the system and be adapted to pinpoint the problem. This way, the problem could even be fixed without sending the engineer over, or at least the engineer would have a basic understanding on what the problem is and what to take with him onsite.

However, not all of the development has to be directly in improving and optimizing the framework. Recalling the layered structure of the tool, see Section III, the monitoring services and test services for certain parts are reusable in different implementations. Hence, the utilization of the Probe Framework and the creation of the monitoring and test services can be perceived as development. As more services are created to the monitoring and test layers, the easier it gets to use the tool. The result is that the effort of deploying the Probe Framework in other systems is lower. For future work, in addition to improving the PF and creating new services, the analysis of the data collected by the instrumentations and the ways it could dynamically guide further instrumentations and testing are of great interest.

For further details on the Probe Framework tool and instrumentation methods [36] offers an in depth view.

Acknowledgment

The authors would like to thank Juha Vitikka from VTT for his collaboration in designing the binary format used in storing the collected traces.

References

[1] M. Pollari and T. Kanstrén (2009) *A Probe Framework for Monitoring Embedded Real-time Systems*. In Proceedings of the Fourth International

Conference on Internet Monitoring and Protection, Venice, Italy, pp. 109–115.

[2] T. Kanstrén (2008) *A Study on Design for Testability in Component Based Embedded Software*. In Proceedings of the Sixth International Conference on Software Engineering Research, Management and Applications, IEEE Computer Society, Prague, Czech Republic, pp. 31–38.

[3] Framework for Dynamic Analysis and Test, Website, [accessed 23.5.2010], http://noen.sf.net

[4] A. Hussain, G. Bartlett, Y. Pryadkin, J. Heidemann, C. Papadopoulos, and J. Bannister (2005). *Experiences with a continuous network tracing infrastructure*. In Proceedings of the ACM SIGCOMM Workshop on Mining Network Data, Philadelphia, Pennsylvania, USA, August 26 - 26.

[5] S. Elbaum and M. Diep (2005) *Profiling Deployed Software: Assessing Strategies and Testing Opportunities*. IEEE Transactions on Software Engineering 31, pp. 312-327.

[6] Y. Bao, M. Chen, Y. Ruan, L. Liu, J. Fan, Q. Yuan, B. Song, and J. Xu (2008) *HMTT: a platform independent full-system memory trace monitoring system*. In Proceedings of the ACM SIGMETRICS international Conference on Measurement and Modeling of Computer Systems, Annapolis, MD, USA, June 02 - 06.

[7] K. Yamanishi and Y. Maruyama (2005), *Dynamic syslog mining for network failure monitoring*. In Proceedings of the Eleventh ACM SIGKDD international Conference on Knowledge Discovery in Data Mining, Chicago, Illinois, USA, August 21 - 24.

[8] M. Diep, M. Cohen, and S. Elbaum (2006) *Probe Distribution Techniques to Profile Events in Deployed Software*. In Proceedings of the 17th International Symposium on Software Reliability Engineering, IEEE Computer Society, Raleigh, NC, USA, pp. 331-342.

[9] H. Giese and S. Henkler (2006) *Architecture-Driven Platform Independent Deterministic Replay for Distributed Hard Real-Time Systems*. In Proceedings of the ISSTA workshop on role of software architecture for testing and analysis, ACM, Portland, Maine, USA, pp. 28-38.

[10] B. M. Cantrill, M. W. Shapiro, and A. H. Leventhal (2004), *Dynamic Instrumentation of Production Systems*. USENIX annual technical conference, Boston, MA.

[11] Mac OS X. Website, [accessed 22.6.2009] http://www.apple.com/macosx

[12] M. Desnoyers and M. Dagenais (2008), *LTTng: Tracing across execution layers, from the Hypervisor to user-space*. Ottawa Linux Symposium.

[13] F. Eigler (2006), *Problem solving with systemtap*. Ottawa Linux Symposium.

[14] SystemTap. Website, [accessed 22.6.2009] http://sourceware.org/systemtap

[15] P. Tuuttila and T. Kanstrén (2008), *Experiences in Using Principal Component Analysis for Testing and Analysing Complex System Behaviour*. In Proceedings of the 21st International Conference on Software & Systems Engineering and their Applications, Paris, France.

[16] Apache log4j. Website, [accessed 22.6.2009] http://logging.apache.org/log4j

[17] C. Lonvick (2001), *The BSD Syslog Protocol*. RFC Editor.

[18] H. Thane and H. Hansson (1999) *Handling Interrupts in Testing of Distributed Real-Time Systems*. In Proceedings of the Sixth International Conference on Real-Time Computing Systems and Applications, IEEE Computer Society, Washington, DC, USA, p. 450.

[19] F. Buschmann, R. Meunier, H. Rohnert, P. Sommerlad, and M. Stal (1996) *Pattern-Oriented Software Architecture: A System of Patterns*. John Wiley & Sons, New York, NY, USA, 476 p.

[20] R.V. Binder (1994) *Design for testability in object-oriented systems*, In Communications of the ACM, Vol. 37, No. 9, Sept. 1994, pp. 87-101.

[21] J. Wegener,H. Sthamer. B. F. Jones, and D. E. Eyres (1997) *Testing real-time systems using genetic algorithms*. Software Quality Control 6, pp. 127–135.

[22] A.S. Tanenbaum (2008) *Modern Operating Systems*. 3rd edition, Prentice Hall, 1104 p.

[23] B. P. Douglass (1999) *Doing Hard Time: Developing Real-Time Systems with UML, Objects, Frameworks, and Patterns*. Addison-Wesley, 749 p.

[24] B. Broekman and E. Notenboom (2002) *Testing Embedded Software*. Addison-Wesley, 348 p.

[25] G. Buttazzo (2006) *Research Trends in Real-Time Computing for Embedded Systems*. ACM SIGBED Review 3, pp. 1–10.

[26] S. Schulz, K. J. Buchenrieder, and J. W. Rozenblit (2002) *Multilevel Testing for Design Verification of Embedded Systems*. IEEE Design & Test of Computers 19, pp. 60–69.

[27] P. B. Menage (2007) *Adding Generic Process Containers to the Linux Kernel*. Ottawa Linux Symposium.

[28] W. Cohen (2005) *Figure 1. Flow of data in SystemTap*. Website, [accessed 22.6.2009] http://www.redhat.com/magazine/011sep05/features/systemtap

[29] IEEE std. 1003.1b-1993 (1993) *POSIX .1b: Real-time Extensions*.

[30] RTAI - the RealTime Application Interface for Linux. Website, [accessed 22.6.2009] https://www.rtai.org

[31] M. Jaakola (2008) *Performance Simulation of Multi-processor Systems based on Load Reallocation*. Masters thesis, Oulu University, Department of Electrical and Information Engineering, Oulu.

[32] J. Vincent, G. King, P. Lay, and J. Kinghorn (2002) *Principles of built-in-test for run-time-testability in component-based software systems*. Software Quality Control 10, pp. 115-133.

[33] UnitTest++. Website, [accessed 22.6.2009] http://sourceforge.net/projects/unittest-cpp

[34] S. E. Chodrow, F. Jahanian, and M. Donner (1991) *Run-Time Monitoring of Real-Time Systems*. In Proceedings of the Real-Time Systems Symposium, IEEE Computer Society Press, San Antonio, TX, USA, pp. 74–83.

[35] J. Voas and K. Miller (1996) *Inspecting and ASSERTing Intelligently*. In Proceedings of the Fourth European Conference on Software Testing, Analysis & Review.

[36] M. Pollari (2009) *A Software Framework for Improving the Testability of Embedded Real-time Systems*. Masters thesis, Oulu University, Department of Electrical and Information Engineering, Oulu.

Depth Perception Within Virtual Environments: Comparison Between two Display Technologies

Abdeldjallil Naceri*†, Ryad Chellali†, Fabien Dionnet† and Simone Toma*†
*Dipartimento di Informatica Sistemistica e Telematica, Università degli Studi di Genova, Via Balbi 5, 16126 Genoa, Italy
†Telerobotics and Applications dept, Istituto Italiano di Tecnologia, via Morego 30, 16163 Genoa, Italy
{abdeldjallil.naceri, ryad.chellali, fabien.dionnet, simone.toma}@iit.it

***Abstract*—Depth perception is one of the key issues in virtual reality. Many questions within this area are still under investigation including the egocentric distance misestimation. In this paper we describe an experiment confirming distance underestimation from another point of view. The approach we developed is based on a very simple task: subjects had to compare relative depths of two virtual objects. The experiment compared performance using head mounted display and stereoscopic widescreen display to evaluate which visual cues subjects use to estimate depth of virtual objects. To minimize motoric effects, subjects were seated and their estimations were only verbal. Likewise, to avoid the well known effects of apparent size, namely the size-distance invariance, the experiment was also performed with conflict sequences: the presented objects had the same apparent sizes with different depths or the same depth but different physical sizes. The obtained results show significant differences between the two devices and confirm the *distance misestimation* phenomenon for head mounted display. Moreover, changing the background color or the shape of the presented objects also had an influence on subjects' performance.**

***Keywords*-Virtual reality, Human-machine interaction, Cognition, Depth perception, Head mounted display, Widescreen display.**

I. Introduction

Immersive viewing devices are key elements for virtual reality [1]. In the paper entitled *The Ultimate Display*, Sutherland [2] depicted a futuristic vision of synthetic worlds and the ways that humans experience virtual realities within these worlds. Nowadays, Sutherland's prophecies are widely spread. Virtual reality (VR), specifically virtual environments (VEs), are used in several areas to support sensitive and complex topics such as psychology, robotics, education, medical therapy and diagnosis, archeology, geography, neuroscience, etc. These research and application fields take advantage of VE capabilities. VE technologies offer flexibility and support innovations by allowing users to explore artificial environments with unconventional rules and interact with virtual objects.

Historically, this field started with computer graphics and 3D representations; VR was mainly concerned with visual channel. With the fast technological advances of the '80s, VR and VE systems began to address other senses. Haptic, tactile, vestibular and auditory sensory channels were introduced to mimic the human sensory system. Since that time the targeted goal has been being improved realism and increased immersion and sense of presence.

In other words, the goal is for users to experience realistic artificial worlds by providing plausible and coherent stimulation and allowing them to interact with these worlds and its objects. This goal has not yet achieved, for a variety of reasons, despite great advances in VE technologies.

Regarding the visual channel, to generating a realistic representation of the real world is very complex and requires a lot of computing resources. The image formation on human retina and its interpretation by human brain is not fully understood. A significant amount literature has been published on visual realism, however the contribution of the visual channel in immersion feeling is not well quantified and few studies have tackled this issue. Slater [3] showed that visual perception is affected by other sensory data streams including kinesthetic proprioception, motoric actions, sounds, etc. For example, a visual flow generated for walking motion increases immersion and presence within VE.

This visual flow phenomenon illustrates the unstable equilibrium of the perception process. This latter is based on a cross-modalities scheme where realism and consequently behavior are affected in an unpredictable way if any of the modalities is itself affected. Because VE systems that are capable of full, accurate simulation are not yet a reality, one can infer that creating such systems is not a trivial task. Consequently VEs rely on actions, interactions and feelings that are distorted, biased and/or incomplete causing malfunctions (fatigue and stress) and biases (physical misestimations). For visual realism the same observation is true: it depends on parameters which are not well identified nor well understood (for a review about the visual cortex and the binocular depth perception see [4]) and any defect or distortion of the visual features can have unexpected effects.

For example, Zago and colleagues [5] tested the validity of the internal model of gravity. They simulated a virtual falling ball displayed on a desktop computer display and a stereoscopic widescreen display (SWD), and subjects adapted to the desktop computer display but not to the SWD. Moreover, in another work to check the same hypothesis, Senot and colleagues [6] investigated the relative role of visual and non-visual information on motor-response timing in interception tasks. Subjects were asked to hit a virtual ball with a virtual racket using a keyboard while the scene was displayed through a head mounted display (HMD). Authors reported that the task was difficult and recorded a success rate lower than 50%. Therefore, these experiments raise the following two questions:

- Why do subjects adapt to desktop display and not to the SWD?
- Why was the task more difficult when using the HMD?

These questions suggest two additional questions about the nature of the obtained results in the previous experiments.

- Does the VR setup introduce a bias?
- Does the VR system distort data?

These questions are generic ones and the previous example is a showcase. Indeed, technological factors modify the behavior of users and consequently affect the understanding of a phenomenon (gravity internal model) not fundamentally concerned with perceptual schemes. More generally, display technologies and interfaces may introduce distortions and biases which are hard to identify and manage. Both users and developers of VR technologies must be less naive and be better informed about current VR limits.

In our work, we aimed to study the effects that visual display technologies can have on a very simple perceptual task. Namely, we wanted to compare SWD and HMD technologies in order to determine how they affect users' behavior.

This research is a part of a lager project to build a VR-based telerobotics system. For these systems, perceiving accurately and correctly remote environment is crucial. A fortiori, the visual perception must provide an exact replica of the remote reality with all corresponding and needed visual cues to ensure natural and coherent sensory motor-based tasks like grasping, reaching or obstacles avoidance and navigation. The replica goal is a theoretical one and we know that it is not reachable with the current technology. Nevertheless, by knowing the limitations of both SWD and HMD, we will be able to prevent inappropriate uses of these display technologies.

The paper is organized as follows. In Section II we give some entry points to research works concerned with visual perception and display technologies. In Section III we describe the designed protocol, the environment conditions, and detail the hardware and software we used. We present in Sections IV and V the obtained results and statistical analysis. In Section VI we discuss our findings concerning depth misestimation issues within VEs and way to exploit them.

II. Background and related work

Even after a decade of research, *visual space perception* remains an active topic with a lot of ongoing work and research efforts. This indicates the importance of the topic and the challenging problems it raises. In this section, we review fundamental notions about human depth perception through vision. We describe the display technologies used in the present work. Additionally, we introduce some techniques used to assess the effectiveness of perceiving distances in VEs. Finally, we discuss the issue of depth misestimation in VR.

A. Visual cues and depth perception

Depth perception results from the integration of several visual cues. Research in this area identify the visual features, the individual visual cues and the ways these cues are processed by the human visual system and combined within human brain to create a vivid three-dimensional perceptual world [7].

From a functional point of view, researchers are still working to understand some fundamental issues including the mapping between real space and its mental representation (visually perceived space), the connection between visual space and motor actions, and the contribution of each visual cue in the building process of the visually perceived space [8][9][10].

From a more basic geometrical point of view, visual cues are 2D entities obtained by a central projection of 3D features. With one eye (one projection), one has monocular cues, including perspective, motion parallax, optic flow, occlusions, lighting, shading, accommodation, etc. When two projections are combined, humans perform an oculomotor convergence-accommodation process and use disparities to deduce the 3D representation of the observed scene.

Both monocular and binocular cues can be characterized by the following two subsets:

- Geometrically and graphically based cues which can be produced by any computer graphics framework such as OpenGL,
- Oculomotor based cues: accommodation and convergence.

As mentioned before, the fusion process is not well understood. Nevertheless, some findings give partial explanations regarding the integration process. Oculomotor and stereopsis cues are known to be inter-dependent (disparity and stereo-acuity), however this dependency is variable. One cue can dominate the other two functions of the distance between observer and the observed object [11]. Depth acuity has been shown to be high within the peripersonal space, defined as the space that can be reached by our hands, typically 1m or less, and low for the extrapersonal space [11][12]. Others [13] have defined these spaces differently using 5m as the defining limit, based on the fact that oculomotor factors become negligible beyond this point. In our experiments, we defined the peripersonal space as less than 1m, which is within both definitions.

B. SWD and HMD visual display technologies

There are two main types of implementations of an immersive virtual display. The first consists of projection widescreens, loud speakers, shutter glasses and/or polarized 3D glasses to provide a stereoscopic stimulation, allowing the user to observe the VE not as a projection on 2D surface, but as 3D solid structures within the experimental site. The second, and the most widely used involves the use of the HMD. With both HMDs or SWDs, binocular visual imagery provides convergence and retinal disparity cues that contribute to the perception of egocentric and allocentric distances in depth.

C. Methods used to assess depth perception cues

Depth and distances are euclidean quantities expressed in meters. Measuring effectiveness of depth cues is equivalent to establishing a relationship between a visual cue (or a set of visual cues) and an euclidean distance. Unfortunately, this input-output scheme is not so valid because humans are weak “instruments” for measuring distances [14][15][16]. In fact, depth perception is considered to be a process leading to an invisible cognitive state [7] and thus inaccessible directly. To overpass this limitation and to take advantage of the human ability to compare, researchers use allocentric or egocentric distances comparison to assess depth cues [8][9][10][17]. In the first case, the observer compares the respective distances between objects and a reference point. In the second case, the observer compares distances between objects relatively to himself. Using this strategy, one can indirectly access the visual cues involved in depth perception process.

In general, there are three methods used to judge egocentric distances: *verbal answers*, *perceptual matching*, and *action-based tasks* [7][10]. In the *verbal answer* case, subject verbally report the comparison between two perceived locations. The task is a forced-choice test and subjects must give answers such as “near” or “far”; “same depths” or “different depths”, etc.

For *perceptual matching* tasks, the subjects directly act on the objects position through interfaces such as a mouse, keyboard or a joystick in order to move the target to a position that matches a reference object [15][18][19].

Finally, for *action-based tasks*, observers are asked to indicate the perceived distance while performing a physical action [7][20][19][21][22]. *Blind walking tasks* is the most common action-based task [7][10][19][20] in which the observer perceives an object at certain distance, then walks with covered eyes until they reach the perceived egocentric distance. This method’s limitation is that large errors are made when subjects attempt to estimate distances beyond 10m [23][24]. Thomson [23] attributed these larger errors to a decay of the spatial memory for target position while walking to the target.

D. Depth misestimation in VE

The visual realism components is highly related to absolute distance judgment or depth perception. It is one of the major issues in VR because rapid understanding and accurate motor-based interactions (like grasping, reaching, intercepting or pointing) in virtual 3D depends on depth perception. Visual richness is positively influenced by monocular cues like dynamic shadows, textured objects, motion parallax, etc. and as visual richness improves so does motor-based interactions [25]. That said, a recurrent issue in both poor and rich VEs is distance misestimation [26]. Systematic underestimation of distances was seen when HMDs were used compared to the same estimation in the real world [20][27]. Similarly, studies on distance perception using the HMD [16][28][29] have found significant underestimation of egocentric distances.

Depth misestimation seems to be a mystery that many authors have proposed explanations for the literature includes several hypotheses to explain the phenomena of depth misestimation in general, and the egocentric underestimation using HMD in particular. In [20], underestimation phenomena was not attributed to the limited field of view of a user while using the HMD. Willemsen and colleagues [30] argue that mechanical properties play a role in the underestimation phenomena. Other explanations include a lack of graphical based-realism does not causes that phenomena [29] or mismatches between the viewed world and the experimental site (e.g., subjects are aware that the viewed scene does not correspond to the place where the experiment is performed) cause the phenomena of distance misestimation [31]. Similarly, some researchers have shown that other factors like visual cues (such as accommodation and convergence) and situations (visually directed actions) may affect distance or depth estimations [22]. Other studies revealed that the misestimation of depth also exists when SWDs are used [32].

In summary, the following hypothesis have been proposed in the literature to explain the misestimation of egocentric distances:

- the reduction of the field of view;
- the weight of the HMD;
- the difference between the viewed world and the experimental place;
- but that monocular versus stereo viewing does not cause it;
- this effect exist in VR when is displayed in SWD.

The egocentric distances perception in VR immersive displays is not fully understood. Moreover, the identification of sources leading to then distance misestimation effects in VE remains an open question, and no specific study has been conducted to directly compare depth perception performances between the two main display technologies.

The first purpose of this our to compare and to evaluate human depth perception in VEs using HMD and SWD. The second point is to confirm whether the depth misestimation in VEs is a reality. We consider this work as a necessary step before attempting to understand and determine the factors underlying this phenomena.

III. Experimental Protocol and Setup

The aim of our experiment is twofolds:

- Determine the effect of display technologies on depth perception
- Study the effects of some visual cues on depth perception

To this end we ran the same experiment using both SWD and HMD. In addition, we varied the background and the shape of the used objects.

In the following we detail our protocol including the used stimuli, hardware and participants.

A. Protocol, stimuli and expectations

The designed experiment was inspired by the works described in [33] and [34] who respectively assessed depth perception in the real world and correlation between size and distance estimation. For both tasks, subjects verbally answered forced choice questions concerning depths of two objects.

In our experiment, we constructed a set of four comparisons types. Each comparison is composed of two stimuli that differ in size and/or in position as reported on Figure 1:

- *Comparison I:* both objects are the same size, but at different depths (apparent sizes are different),
- *Comparison II:* both objects are the same size, and at the same depth (apparent sizes are equal),
- *Comparison III:* both objects are different sizes and at the same depth (apparent sizes are different),
- *Comparison IV:* both objects are different sizes, but at different depths (such that apparent sizes are equal),

This set of pair-comparisons takes into account the findings of [33] and [34]. In the first paper it was found that an object which was consistently overestimated in size was consistently overestimated in distance (size-distance paradox) which strongly support the hypothesis about the size-distance covariance in depth estimation. In addition, Berryhill and colleagues [34] recently reported a high degree of accuracy by healthy subjects in judging either the size or the distance of real objects even though both perceptive variables were not covarying and were the only cues available.

Naively, the comparisons I and II might be sufficient for our experiment, however, subjects could potentially rely on apparent size which could make a difficulties in understanding the results: the ability to discriminate between those

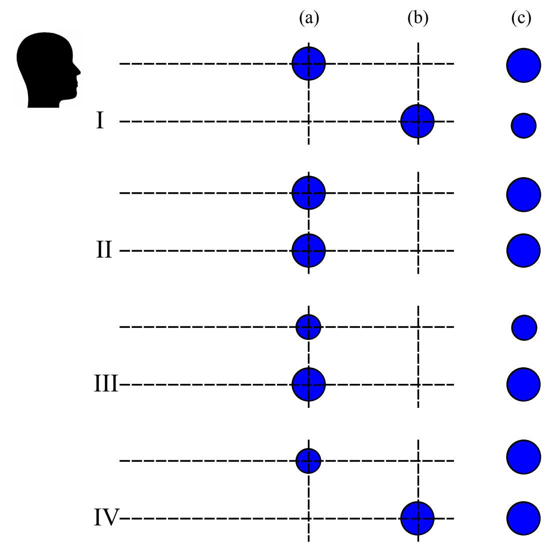


Figure 1: Four comparisons of two objects, (a) and (b) represent the virtual objects presented to subjects and (c) corresponding retinal size

perceiving depth correctly and those relying on apparent size to give a correct answer. Therefore, two other comparisons were added, III and IV. These comparisons amplify the size/distance conflict leading a higher ambiguity for subjects relying on apparent size. To limit the visual cues subjects could apply during the task, we used an impoverished environment (uniform background). Likewise, lighting and shading conditions remained constant. We also used a spherical object to force subjects to rely only on stereopsis and convergence. Finally, the stimuli were displayed directly in front of the subjects to avoid any parasitical motor activity (head movement).

The second part of the experiment dealt with the effect of some visual cues on depth estimation. To that purpose we varied both background and object shapes. For the backgrounds, we used two uniform colors, black and white (BB and WB). These conditions were strictly followed and applied for both SWD and HMD.

B. Task and conditions

1) Objects: The presented objects were two spheres of diameters 7.5cm and 10cm and two cubes of edge lengths 7.5cm and 10cm as shown on Figure 2.

2) Scene organization: The virtual objects were displayed exactly 60cm and 80cm in front of subjects' eyes. These distances were chosen as a trade-off imposed by two opposite constraints. First that the virtual objects be within the subject's peripersonal space because the present work is not only focused on depth perception but also is the first step in more complex study dealing with motor action during reaching tasks. Nevertheless, virtual objects cannot

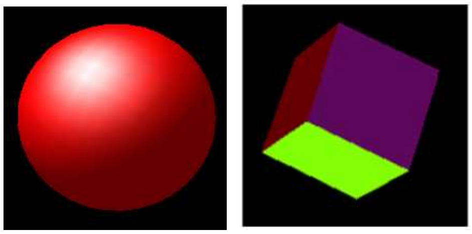


Figure 2: Displayed virtual sphere and cube

be displayed too close to the subject without creating visual discomfort and stress.

3) Timing: Each object was individually presented to avoid the apparent motion cues that might alter subjects' perceptual judgments. The first for 3s, following by a 2s pause without virtual objects before the second object was presented for a total of 8s. These 8s sequences were repeated 10 times (i.e., 80 trials per conditions described in Figure 1). Both the order of virtual objects presented (sphere/cube) and use of HMD/SWD was randomized. Likewise, the order of the conditions was randomly assigned to avoid bias due to learning process. The each experimental condition took approximately 20 min with a brief pause at the half way point.

4) Question: For all conditions, two alternative forced choice (2 AFC) was used. Subjects were asked to verbally answer the following question after each comparison of the two displayed virtual objects: "*Are the two objects you just saw located at the same position?*" They had to answer "*yes*" or "*no*" before the next comparison started.

C. Hardware

The experiment was carried out in TEle Robotics and Applications department (TERA) VR room.

The SWD system includes two videoprojectors, two polarization filters and one widescreen. The videoprojector model is the evo22sx+ from Projectiondesign with luminance of 3000 ANSI lumens. They are placed side-by-side on the ceiling of the VR room and are equipped with two orthogonal circular polarization filters. Both beams are oriented to the widescreen and the projection distortion is corrected until getting two perfectly overlayed rectangular images of dimensions 1.805×1.535 m^2. The screen is polarized such that wearing light passive polarized glasses allow subjects' eyes to see two different images (Figure 3).

The HMD is a binocular Cybermind Visette45. The optical system designed for subjects' eyes to accommodate on a plane located 2m in front for a diagonal field of view of $45°$. It weighs approximately 750 g (Figure 4).

In both cases, left and right images of resolution 1280×1024 were generated using the OpenGL library on a PC Dell bi-Xeon 3GHz with 8Gb of RAM running GNU/Linux and displayed simultaneously at 50Hz. The subjects were

Figure 3: A subject wearing polarization filter glasses and head tracking device performing the task in SWD presentation

Figure 4: A subject performing the task in HMD presentation

seated approximately 1.8m in front of the projection screen (see Figure 3) in order to allow them to accommodate approximately at the same distance for both display device. The subjects were asked to maintain a fixed position and to move as little as possible during the trials of the experiment.

D. Participants

Eight observers took part in the study, five males and three females of age 42 ± 10 years old. All were naive to

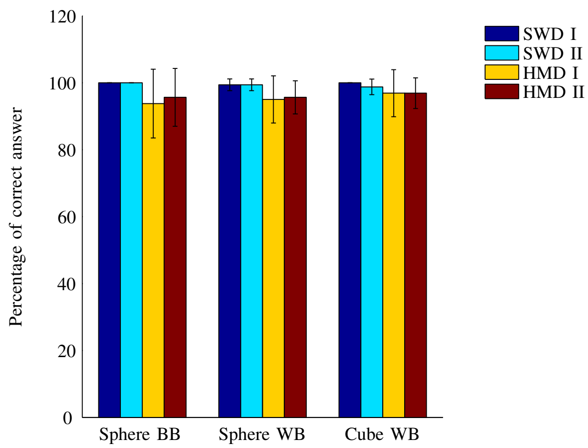


Figure 5: Average rate of correct answers in SWD and HMD conditions considering the comparisons I & II

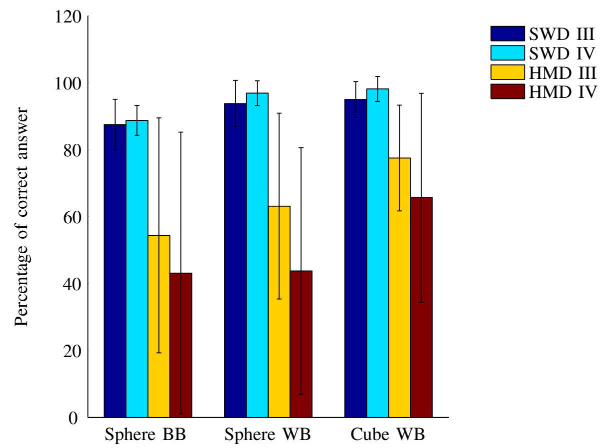


Figure 6: Average rate of correct answers in SWD and HMD conditions considering the comparisons III & IV

the purpose of the experiment and had normal or corrected to normal visual acuity.

IV. Preliminary analysis of the results

In this section we present the obtained results with respect to the two main questions:

- SWD vs HMD comparison
- Effects of specific visual cues on depth perception

In the first part we present subjects' success rates in different conditions using SWD and HMD. To that end, we start with a global analysis based on mean and standard deviation. To refine, an ANOVA was completed to determine whether there were any significant effects regarding presentations, conditions and/or subjects' personal performances.

A. Global analysis

1) SWD vs HMD: To compare depth perception performances in both SWD and HMD conditions, we calculated means and standard deviations for the four comparisons (see Figures 5 and 6). For the SWD presentation, subjects showed good performances in all comparisons. However, we noticed larger variability in the comparisons III and IV. In the HMD presentation, the subjects' rates of success were greater than 85% in the comparisons I and II. For comparisons III and IV, performances were lower than 75%. In addition standard deviations for these two comparisons were larger than any other comparison.

2) Effects of object's shape and background: For the SWD, we noticed that there was no specific effect due to the background nor the object shape for comparisons I and II. For comparisons III and IV, the success rate gradually increased (black background $<$ white background $<$ cube). For the HMD, the same tendency although with a greater slope. In other words, the enhancement effect was more pronounced with HMD.

B. ANOVA analysis

ANOVA tests revealed that the order of presentation within the four comparisons had absolutely no effect ($p > 0.05$). Thus, in all analysis we used four comparisons instead of the eight sub-comparisons of the designed protocol and effectively performed during the experiment.

1) SWD vs HMD: When we compared the results obtained for the HMD compared with those of SWD, ANOVA tests revealed significant differences, particularly for the comparisons III and IV (see Table I).

2) Effects of object's shape and background: A one-way ANOVA test for different conditions reveals no significant difference between the three comparisons I, II, and III for the three conditions in the SWD presentation (I: $F[2,21] = 1$ and $p > 0.39$, II: $F[2,21] = 1.11$ and $p > 0.35$, III: $F[2,21] = 2.89$ and $p > 0.08$). As for comparison IV, the ANOVA test revealed a significant difference between the "sphere-black background" condition compared to the "sphere-white background" and "cube-white background" conditions ($F[1,14] = 15.77$ and $p < 0.0014$, $F[1,14] = 21$ and $p < 0.0001$).

3) Variability between subjects: The ANOVA test revealed a significant difference between subject performances only for the HMD condition in comparisons III and IV (III: $F[7,23] = 7.14$ and $p < 0.00001$, IV: $F[7,23] = 16.29$ and $p < 0.000001$). In fact, there was one subject who had good performances in both presentations SWD and HMD. Furthermore, two subjects had an average rate of success more than 50% in the HMD condition for the comparisons III and IV but less compared to their performances in the SWD. For the remaining subjects, had a noteworthy decrease in the accuracy rates for the HMD condition in all stimulus comparisons.

ANOVA test revealed no significant difference between comparisons III and IV for the HMD presentation. One can

Table I: ANOVA tests result for the four comparisons across three conditions including F-values and p-values, comparison between SWD and HMD.

ANOVA test	Comp.	Conditions		
		Sphere BB	Sphere WB	Cube WB
p-values	I	$p > 0.11$	$p > 0.11$	$p > 0.23$
	II	$p > 0.11$	$p > 0.06$	$p > 0.32$
	III	$p < 0.02$	$p < 0.01$	$p < 0.01$
	IV	$p < 0.01$	$p < 0.001$	$p < 0.01$
F-value: F(1,14)	I	2.97	2.88	1.58
	II	2.97	4.06	1.07
	III	6.81	9.16	8.79
	IV	9.30	16.49	8.55

see from the Figures 5 and 6 that the cube condition was better than the others two conditions for the HMD in the comparisons III and IV. This results were confirmed by using the d' method in the next section.

V. Results and Data Analysis within the SDT framework

In our first analysis revealed a large variability between subjects. ANOVA tests are not reliable enough in such situations. To overcome this limitation and to more thoroughly analyze the data, we choose the SDT framework to refine the analysis and to better understand the found results.

A. SDT description

Signal detection theory (SDT) is used to analyze experimental data with categorization tasks using ambiguous stimuli. Tanner and Swets [35] proposed a statistical decision theory and specific ideas about electronic signal-detecting devices to build a model that closely approximates of how people actually behave that in a such situations. The model was described in detail and named "*signal detection theory*" by Green and Swets [36].

We choose this framework to evaluate and analyze our data because the task is based on ambiguous stimuli. Moreover, the data in our experiments are binary answers,"*yes*" and "*no*". The SDT-based analysis can provide estimations of subjects capabilities in terms of discriminative behavior or sensitivity regarding the presented stimuli d' [36].

B. SDT in our experiment

The task in our experiment is to judge whether two stimuli in comparison are at the same depth ("*yes*" or "*no*" answers). Let us consider the comparisons as new stimuli or meta-stimuli (the presentation of two successive objects within the same comparison) to be analyzed with the SDT. Disambiguation between two meta-stimuli may be related to relative apparent sizes and depth displayed during each comparison. For instance, for comparison I and III, we have the same configuration regarding apparent sizes and different depths configuration (Figure 1). Likewise, we have the inverse situation in comparisons II and IV: the same configuration for apparent sizes and different depths. The disambiguation and the consequent indirect questions that we used in the SDT are the following:

- Given different apparent sizes, are the depths judged to be equal or not? (disambiguation between I and III)
- Given the equal apparent sizes, are the depths judged to be equal or not? (disambiguation between II and IV)

Correct estimations judgments in the previous two conditions means that subjects overcome the size-distance paradox and rely only on the actual depth perception they have. In the remainder of the section, the meta-stimuli word is replaced by stimuli. If a "*yes*" answer to a presented stimulus is a correct answer, it is called a hit (H); but if a "*yes*" answer to a stimulus is a mistake, it is called a false alarm (FA). If a "*no*" answer is the correct response, it is called a correct rejection; but if a "*no*" answer is incorrect, it is called a miss. The proportions of hits and false alarms reflect the effect of two underlying parameters. The first parameter reflects the separation between the comparison (e.g., I) and the ambiguous comparison (e.g., III) of the stimulus. The second parameter is the strategy of the participants. The expected SDT models are expected to quantify subjects' perceptive sensibility in detecting environmental changes. Specifically, the performances when using SWD versus HMD devices and the effects of changing the background or the object shape.

Individual d' values were extracted from the differences between the normalized percentage of correct hit answers and the normalized percentage of false alarm answers. The hit rate is simply the proportion of "same apparent size and different position" responses that occurs for comparison IV. The false alarm rate is the proportion of "same apparent size, same position" for comparison II.

C. SDT analysis

In SDT framework, the value of d' is given as follows

$$d' = z_{\mathrm{H}} - z_{\mathrm{FA}}, \tag{1}$$

where z_{H} and z_{FA} are respectively the normalized probabilities of hit and false alarm rates.

In our analysis, we computed the normal distribution for all subjects by using the bootstrap procedure in order to estimate an accurate mean and variation. The aim of this analysis is to clearly observe how the subjects distinguish the comparisons I/II from III/IV respectively. Indeed, we statistically reinforced the obtained data since our set is small and contain a variability in some conditions. The best approach is to apply a bootstap procedure to extract a idealized models and assumptions, as introduced by Efron [37], in order to estimate and approximate a realistic model with normalized Gaussian distributions. The obtained mean values and variances characterize subjects' decision making behavior.

D. SWD vs HMD

The variance of the normal distributions describes the standard deviation of the population obtained by bootstrapping. The accuracy of the subjects' performances can be explained by the different variances characterized the normal distributions (Figures 9, 10). For the obtained representations, we found subjects' performances with SWD are better than those obtained with HMD across all four comparisons. Indeed, the obtained normal distributions related to SWD presentation (Figures 9) had smaller variances were clearly separated compared to those obtained for the HMD (Figure 10).

This latter observation confirms the results given by the one-way ANOVA test. Moreover, SDT framework helped us to quantify the differences between these two presentations. Indeed, the d' mean value for the HMD presentation was approximately half compared to the SWD presentation as shown by Figure 7.

$$d'_{\mathrm{HMD}} \approx 0.5 \times d'_{\mathrm{SWD}} \tag{2}$$

This indicates that there was more confusion and subjects had difficulty distinguishing between stimulus comparisons using HMD.

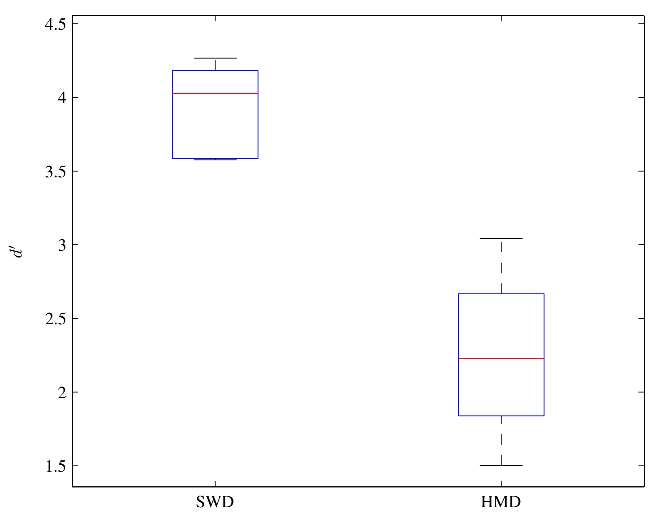


Figure 7: Subjects' d' average for SWD and HMD

E. Object shape effects on depth perception

It is obvious from Figures 9 that varying object shape had no effect on performances. Indeed, the mean values of the normal distributions are identical as shown in Figures 9b and 9c. On the contrary, for the HMD presentation there was clearly a difference between shapes since the two obtained normal distributions were more evidently separated with a cube compared to a sphere (see Figures 10b and 10c).

F. Comparison between pair-comparisons

From Figure 9, we observe that the variances of the normal distributions of comparisons I, II and IV using SWD are very low. This reflects the small variability of the population in these cases. On the contrary, we recorded a higher variability with the stimulus comparison III. Therefore, in the SWD presentation this comparison characterized by different apparent sizes but same position seems to be the most ambiguous to the subjects .

For the HMD presentation, we observed that the variances of the normal distributions are low for comparisons I and II (see Figure 10). As for the two other comparisons, the variability for comparison III is higher than those for comparisons I an II, but lesser than for comparison IV. Thus, the comparison IV, characterized by constant apparent size with different depths, is the most difficult to judge using a HMD.

VI. Discussion

A. Influence of the presentation: SWD vs HMD

The first goal of this work was to determine if observer performance in a depth perception task varies with respect to display technology. Two different statistical methods, ANOVA and d'-based method, revealed a difference. Indeed, while estimating relative depths is almost perfectly achieved for all the four presented comparisons with SWD, this is not true with HMD, especially when ambiguous situations are presented. Indeed, the d' mean value for the HMD presentation is approximately the half compared to the one of the SWD presentation (see Figure 7).

More specifically, for the SWD presentation the observers showed good performances even in the ambiguous comparisons. This contradicts with [32] which reported that subjects misestimate depth even by using widescreen. On the contrary, for the HMD presentation our finding is coherent with several studies showing that observers misestimate egocentric distances in VEs when they wear a HMD [16][28][29]; this remain true regardless of experimental methodology in these studies. Others have shown that distance misestimation in VEs by using HMD is not due to the limited field of view [20]. On the contrary the field of view restrictions of HMDs, in addition to other parameters that constraint head movements such as the weight, may have an influence on the accuracy of distance estimations [27].

After the experiment, all subjects were asked about the strategy they used to achieve the task. In SWD presentation, subjects answered that the task was more realistic and the virtual objects seemed to be reachable by their hands, in other words, the objects were within their peripersonal space. They did not state strategies using apparent size. On the contrary in the HMD condition, they were particularly less accurate in comparisons where apparent size does not reflect the correct depth. During post-experiment interviews, all

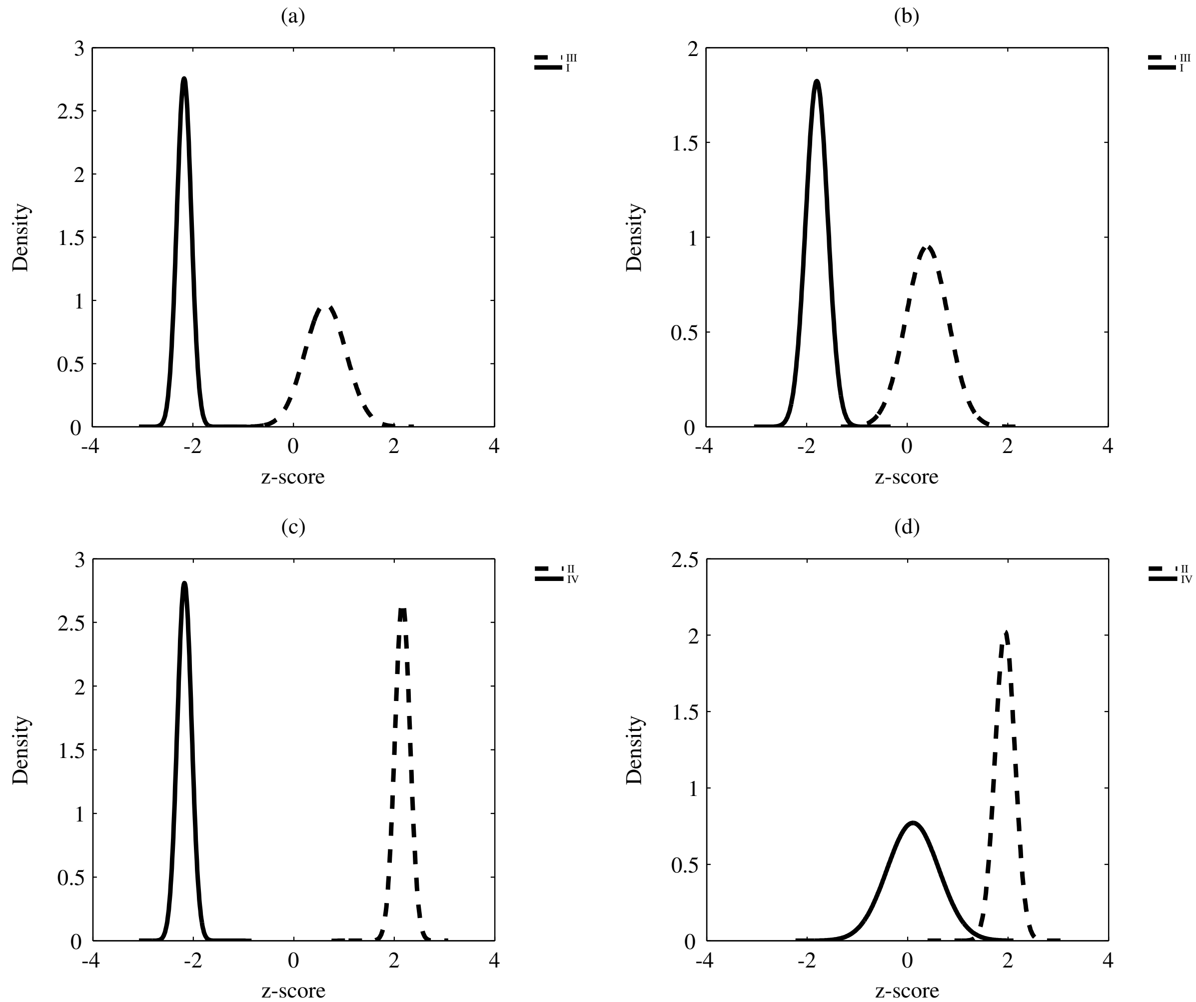


Figure 8: Normal distribution of subjects' z-score for the condition sphere with black background, (a,c) for SWD and (b,d) for HMD

subjects reported that the task was more difficult with the HMD than the SWD.

B. Influence of the background and shapes changes

In the data analysis part, object shape had an influence on subjects performances for the HMD but not for the SWD presentation. Specifically, for the HMD, performances were better with the cube than with the sphere. These differences were particularly visible with the d'-based analysis. This suggests that subjects relied in this case more on disparity cues because the cube contains edges, vertices and perspective that give more information than the sphere for depth evaluation. These latter cues helped subjects estimate depth and overcome ambiguous comparisons in some trials.

Therefore, one can presume that subjects missed some effective cues in the HMD presentation that were present and used with SWD. One hypothesis might be that wearing HMD isolate subjects' visual channel from all other stimuli, leading to misestimate of egocentric distances with regard to the body as reference. By definition egocentric frames of reference based on the body or specific parts of it, to define spatial positions [38][39].

As for background effects, we found that changing color influenced the observers' performance: (comparison IV, see ANOVA test). This finding contradicts reports that showed different VEs conditions did not impact observers' depth estimation [14].

C. Stimulus comparisons

The first analysis shows clearly that the performances of subjects were better for the SWD: the accuracy rates were higher and the variability lower. More explicitly, subjects more effectively resolved conflicts present in stimuli comparisons III and IV. Indeed, they did not rely on angular

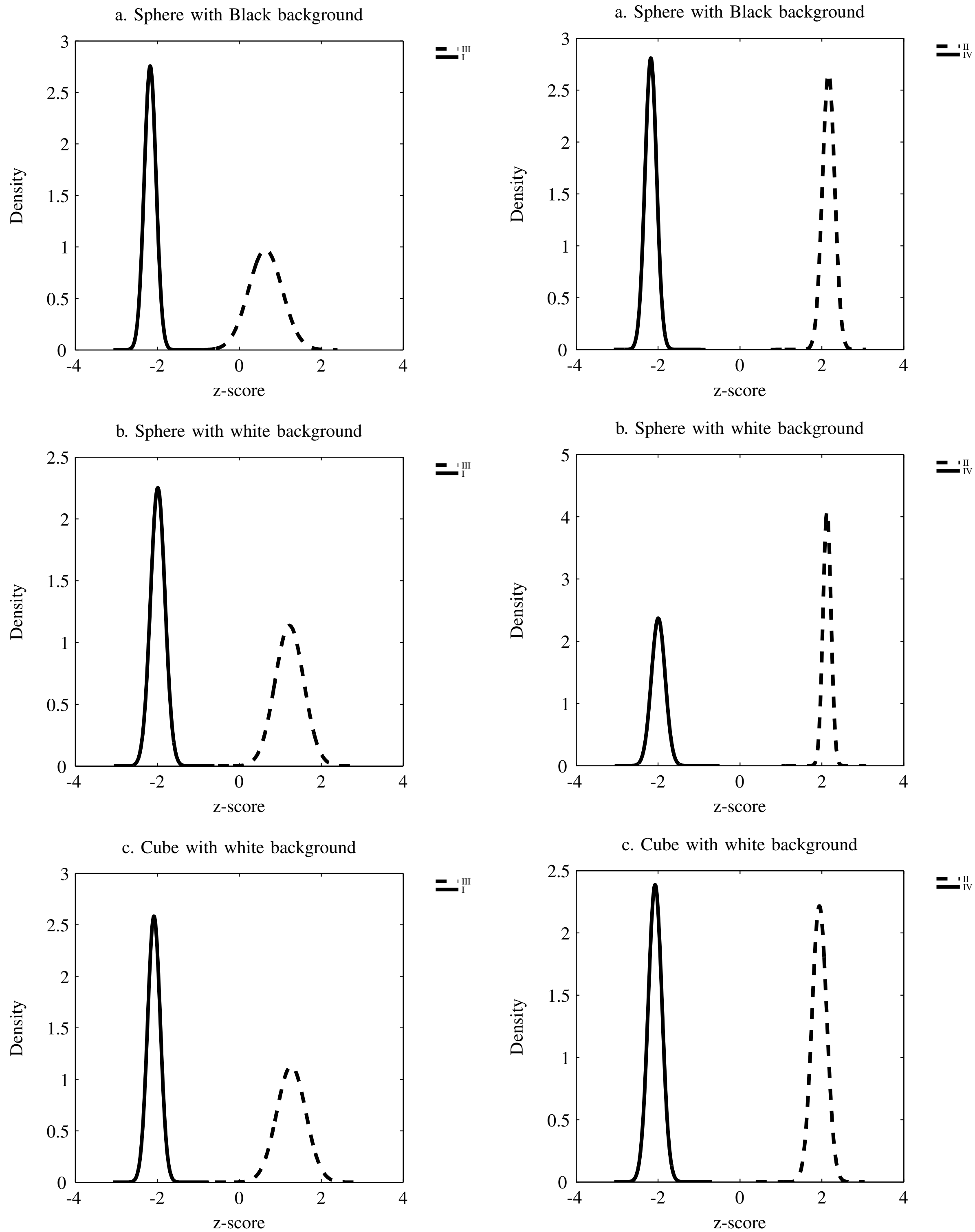


Figure 9: Normal distribution of subjects' z-score for the three conditions using the SWD

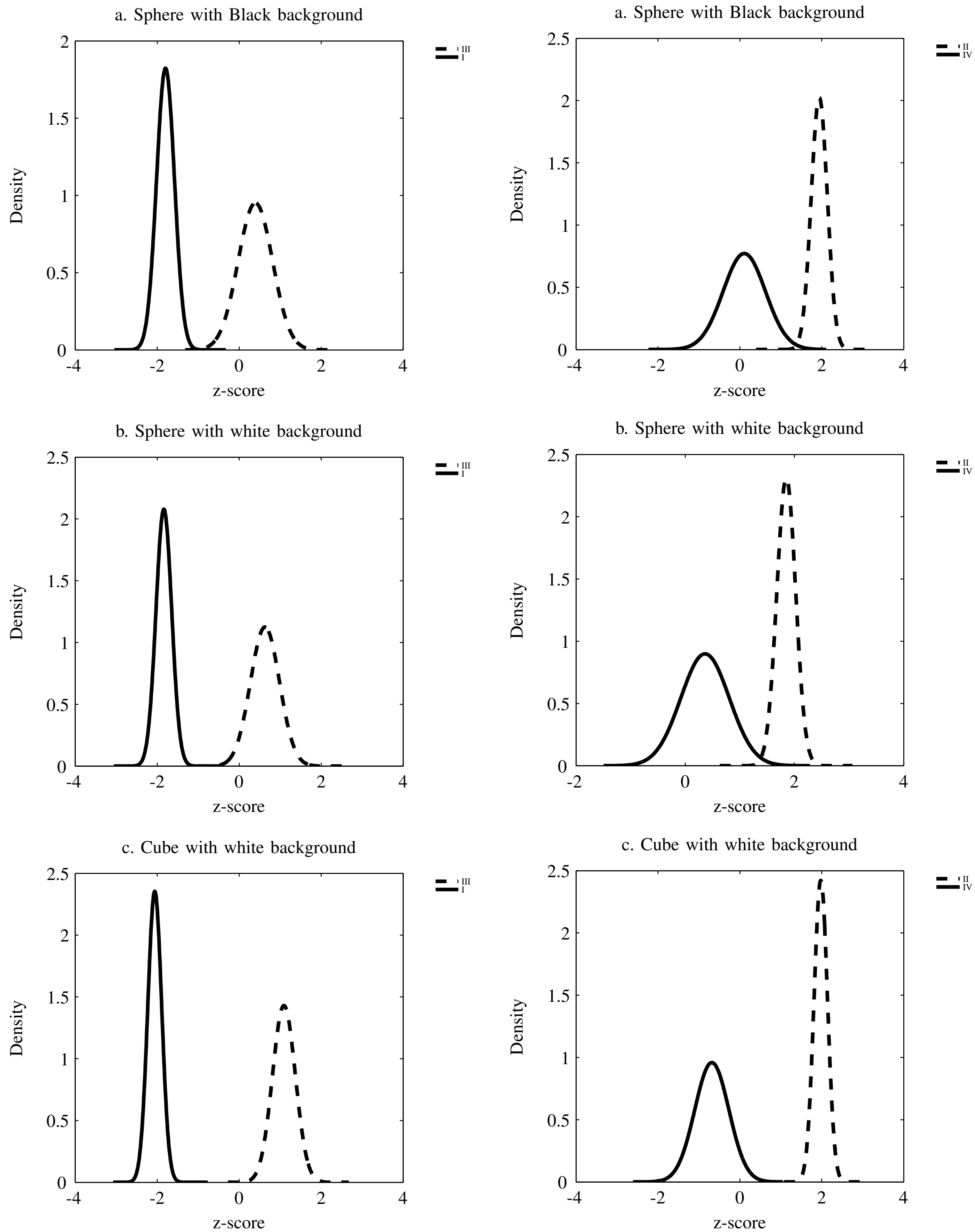


Figure 10: Normal distribution of subjects' z-score for the three conditions using the HMD

size to perceive depths in the SWD comparisons. With the HMD, there was more confusion and ambiguity, especially for the comparisons III and IV. For the comparisons I and II we obtained similar results for both conditions. Furthermore, the effect of the shape of the stimuli had a noticeable impact.

When analyzing the different distributions, it appears that the comparison III was the worst in terms of variance that characterize subject performance for SWD presentation. In this comparison, two objects with different apparent size were presented at the same position. Moreover, the distribution was large and close to zero. This means that subjects react with great variability and most of them answered randomly. This, suggests that subjects do not rely on apparent size nor on the disparity within this type of display. The same phenomena occurred for the comparison IV (same apparent size, different depths) but only when the HMD was used suggesting that in this comparison subjects rely more on the apparent size than on disparity. This fact is confirmed in the cube condition (the disparity is more effective with cube vertex): the performances were less variable and the answers were more accurate.

VII. Conclusion and future work

In this study, we qualitatively evaluated human depth perception in VR systems. This issue is fundamental to many areas particularly for brain and cognitive science research. Indeed, the use of VR has been increasingly used for simulation allowing researchers to create a variety of realistic stimuli under experimental conditions. To fulfill these needs, VR tools must be perfect, or at least well understood, to avoid co-lateral effects and biases, otherwise experimental results and interpretations will suffer from VR-induced distortions and illusions.

Observers were instructed to estimate the depth of virtual objects in four assigned comparison tasks which varied object shape, background color and display technology. Given the relatively small sample size (n of 8) of this study, we realize that care must be taken when drawing statistical conclusions. To address this concern, we confirmed the results obtained through our first statistical analysis by doing a second round of analysis. Specifically, we employed the d' method combined with bootstrap statistics. The d' method allows us to derive additional statistics and provide additional information to confirm or question the original conclusions. The bootstrap method augmented the data in our data set allowing us to overcome the variability observed with our subjects.

More specifically we investigated factors leading users to misestimate the egocentric distances within the peripersonal space when wearing HMDs. To do this, we conducted an experiment comparing human performance on a variety of depth perception tasks when using HMD versus SWD. We had two noteworthy findings. The first is that subjects are able to correctly compare depths using both systems when objects are of the same physical size, however, when objects are of different physical sizes this capability only persisted when subjects used a SWD (performance decreased with a HMD). This allows us to conclude our second finding, which is that subjects rely on apparent size when making depth comparison using a HMD, but not with a SWD. One key difference between the HMD and SWD experience is that a subject is unable to see his or her own body, which suggests that humans may use relevant visual cues from their body's position to judge depth and distance.

The possibility that seeing one's own body provides important visual cues for depth perception will be investigated in future research of this lab. An experimental protocol is currently being developed to that allows researchers to vary the presence of visual cues from subjects' own bodies in the context of a SWD. In other words, we will next explore whether seeing one's own body influences a subject's depth perception abilities with virtual objects while excluding any possible variables that inherently exist between HMDs and SWDs. Additional questions remain regarding the possible importance of physiological properties, specifically accommodation, convergence, or eye movement, that might be investigated with eye tracking technology.

References

[1] A. Naceri, R. Chellali, F. Dionnet, and S. Toma, "Depth perception within virtual environments: A comparative study between wide screen stereoscopic displays and head mounted devices," in *ComputationWorld'09*. IEEE Computer Society, 2009, pp. 460–466.

[2] I. Sutherland, "The ultimate display," in *Proceedings of the International Federation of Information Processing Congress*, 1965.

[3] M. Slater, A. Steed, J. McCarthy, and F. Maringelli, "The influence of body movement on subjective presence in virtual environments." *Hum Factors*, vol. 40, no. 3, pp. 469–477, Sep 1998.

[4] A. J. Parker, "Binocular depth perception and the cerebral cortex." *Nat Rev Neurosci*, vol. 8, no. 5, pp. 379–391, May 2007.

[5] M. Zago, G. Bosco, V. Maffei, M. Iosa, Y. P. Ivanenko, and F. Lacquaniti, "Internal models of target motion: expected dynamics overrides measured kinematics in timing manual interceptions." *J Neurophysiol*, vol. 91, no. 4, pp. 1620–1634, Apr 2004.

[6] P. Senot, M. Zago, F. Lacquaniti, and J. McIntyre, "Anticipating the effects of gravity when intercepting moving objects: differentiating up and down based on nonvisual cues." *J Neurophysiol*, vol. 94, no. 6, pp. 4471–4480, Dec 2005.

[7] J. E. Swan, A. Jones, E. Kolstad, M. A. Livingston, and H. S. Smallman, "Egocentric depth judgments in optical, see-through augmented reality." *IEEE Trans Vis Comput Graph*, vol. 13, no. 3, pp. 429–442, 2007.

[8] J. Loomis, "Distal attribution and presence," *Presence*, vol. 1, pp. 113–119, 1992.

[9] J. M. Loomis, J. A. Da Silva, J. W. Philbeck, and S. S. Fukusima, "Visual perception of location and distance," *Current Directions in Psychological Science*, vol. 5, pp. 72–77, 1996.

[10] J. M. Loomis and J. M. Knapp, "Visual perception of egocentric distance in real and virtual environments," *In L. J. Hettinger and M. W. Haas (Eds.), Virtual and Adaptive Environments*, pp. 21–46, 2003.

[11] J. Cutting and P. Vishton, "Perceiving layout: The integration, relative dominance, and contextual use of different information about depth," *In Epstein, W., & S. Rogers (Eds.), Handbook of Perception and Cognition: Perception of Space and Motion*, vol. 5, pp. 69–117, 1995.

[12] F. H. Previc, "The neuropsychology of 3-d space." *Psychol Bull*, vol. 124, no. 2, pp. 123–164, Sep 1998.

[13] S. Nagata, "How to reinforce perception of depth in single two-dimensional pictures," *Pictorial communication in virtual and real environments*, pp. 527–545, 1991.

[14] C. Armbrster, M. Wolter, T. Kuhlen, W. Spijkers, and B. Fimm, "Depth perception in virtual reality: distance estimations in peri- and extrapersonal space." *Cyberpsychol Behav*, vol. 11, no. 1, pp. 9–15, Feb 2008.

[15] T. Grossman and R. Balakrishnan, "An evaluation of depth perception on volumetric displays," in *AVI '06: Proceedings of the working conference on Advanced visual interfaces*. New York, NY, USA: ACM, 2006, pp. 193–200.

[16] R. Messing and F. H. Durgin, "Distance perception and the visual horizon in head-mounted displays," *ACM Trans. Appl. Percept.*, vol. 2, no. 3, pp. 234–250, 2005.

[17] J. A. Aznar-Casanova, E. H. Matsushima, J. A. D. Silva, and N. P. Ribeiro-Filho, "Can exocentric direction be dissociated from its exocentric distance in virtual environments?" *Percept Psychophys*, vol. 70, no. 3, pp. 541–550, Apr 2008.

[18] S. R. Ellis and B. M. Menges, "Localization of virtual objects in the near visual field." *Hum Factors*, vol. 40, no. 3, pp. 415–431, Sep 1998.

[19] B. Wu, T. L. Ooi, and Z. J. He, "Perceiving distance accurately by a directional process of integrating ground information." *Nature*, vol. 428, no. 6978, pp. 73–77, Mar 2004.

[20] J. M. Knapp and J. M. Loomis, "Limited field of view of head-mounted displays is not the cause of distance underestimation in virtual environments," *Presence*, vol. 13, pp. 572–577, 2004.

[21] S. H. Creem-Regehr, P. Willemsen, A. A. Gooch, and W. B. Thompson, "The influence of restricted viewing conditions on egocentric distance perception: implications for real and virtual indoor environments." *Perception*, vol. 34, no. 2, pp. 191–204, 2005.

[22] M. Mon-Williams and J. R. Tresilian, "Ordinal depth information from accommodation?" *Ergonomics*, vol. 43, no. 3, pp. 391–404, Mar 2000.

[23] J. A. Thomson, "Is continuous visual monitoring necessary in visually guided locomotion?" *J Exp Psychol Hum Percept Perform*, vol. 9, no. 3, pp. 427–443, Jun 1983.

[24] J. Decety, M. Jeannerod, and C. Prablanc, "The timing of mentally represented actions." *Behav Brain Res*, vol. 34, no. 1-2, pp. 35–42, Aug 1989.

[25] J. Gibson, *The Ecological Approach to Visual Perception*. London: Lawrence Erlbaum Associates, 1986.

[26] A. Murgia and P. M. Sharkey, "Estimation of distances in virtual environments using size constancy," *The International Journal of Virtual Reality*, vol. 1, pp. 67–74, 2009.

[27] P. Willemsen, M. B. Colton, S. H. Creem-Regehr, and W. B. Thompson, "The effects of head-mounted display mechanical properties and field of view on distance judgments in virtual environments," *ACM Trans. Appl. Percept.*, vol. 6, no. 2, pp. 1–14, 2009.

[28] B. G. Witmer and P. B. Kline, "Judging perceived and traversed distance in virtual environments," *Presence: Teleoper. Virtual Environ.*, vol. 7, no. 2, pp. 144–167, 1998.

[29] W. B. Thompson, P. Willemsen, A. A. Gooch, S. H. Creem-regehr, J. M. Loomis, and A. C. Beall, "Does the quality of the computer graphics matter when judging distances in visually immersive environments?" 2002.

[30] P. Willemsen, A. A. Gooch, W. B. Thompson, and S. H. Creem-Regehr, "Effects of stereo viewing conditions on distance perception in virtual environments," *Presence: Teleoper. Virtual Environ.*, vol. 17, no. 1, pp. 91–101, 2008.

[31] V. Interrante, B. Ries, and L. Anderson, "Distance perception in immersive virtual environments, revisited," in *Proc. Virtual Reality Conference*, 2006, pp. 3–10.

[32] J. M. Plumert, J. K. Kearney, J. F. Cremer, and K. Recker, "Distance perception in real and virtual environments," *ACM Trans. Appl. Percept.*, vol. 2, no. 3, pp. 216–233, 2005.

[33] H. E. Gruber, "The relation of perceived size to perceived distance," *The American Journal of Psychology*, vol. 67, pp. 411–426, 1954.

[34] M. E. Berryhill, R. Fendrich, and I. R. Olson, "Impaired distance perception and size constancy following bilateral occipitoparietal damage." *Exp Brain Res*, vol. 194, no. 3, pp. 381–393, Apr 2009.

[35] W. P. Tanner and J. A. Swets, "A decision-making theory of visual detection." *Psychol Rev*, vol. 61, no. 6, pp. 401–409, Nov 1954.

[36] D. Green and J. Swets, *Signal detection theory and psychophysics*. Wiley, New York, 1966.

[37] B. Efron, "Bootstrap methods: Another look at the jackknife," *Ann. Statist.*, vol. 7, pp. 1–26, 1979.

[38] K. Ball, D. Smith, A. Ellison, and T. Schenk, "Both egocentric and allocentric cues support spatial priming in visual search." *Neuropsychologia*, vol. 47, no. 6, pp. 1585–1591, May 2009.

[39] G. Committeri, G. Galati, A.-L. Paradis, L. Pizzamiglio, A. Berthoz, and D. Lebihan, "Reference frames for spatial cognition: different brain areas are involved in viewer-, object-, and landmark-centered judgments about object location." *J Cogn Neurosci*, vol. 16, no. 9, pp. 1517–1535, Nov 2004.

Integration of Condition Monitoring in Self-optimizing Function Modules Applied to the Active Railway Guidance Module

Christoph Sondermann-Wölke
University of Paderborn
Mechatronics and Dynamics
Fürstenallee 11
33102 Paderborn, Germany
Email: chrisson@upb.de

Walter Sextro
University of Paderborn
Mechatronics and Dynamics
Fürstenallee 11
33102 Paderborn, Germany
Email: walter.sextro@upb.de

***Abstract*—New mechatronic systems, called self-optimizing systems, are able to adapt their behavior according to environmental, user and system specific influences. Self-optimizing systems are complex and due to their non-deterministic behavior comprise hidden risks, which cannot be foreseen in the design phase of the system. Therefore, modifications of the ISO 17359 condition monitoring policy for being able to cope with this new kind of systems are presented. Besides avoiding critical situations evoked by self-optimization, the proposed concept uses self-optimization to increase the dependability of the system. This concept is applied to the active guidance module of an innovative rail-bound vehicle. First test drives provide information for the enhancement of the implementation of real-time switching to appropriate control strategies. The different control strategies are investigated in detail. It is illustrated that influences on the system like different track sections or the desired velocity of the RailCab effect the system and can lead to a higher amount of flange contacts, which indicate higher wear and thus a reduction of the availability of the system. Therefore, these influences should be minded within the condition monitoring policy. Consequently, this article presents the condition monitoring policy for self-optimizing function modules and its application to the active railway guidance module.**

***Keywords*-dependability; condition monitoring; self-optimization; active railway guidance module**

I. Introduction to self-optimizing systems

Classical mechatronic systems use knowledge of mechanical and electrical engineering as well as information technology to open up new design concepts. Self-optimizing systems, which are the focus of research at the collaborative research center (CRC) at the University of Paderborn, augment these mechatronic systems with intelligent information processing [1][2]. This “inherent intelligence” is able to adapt the objectives of the system and hence the behavior of the system to changing influences. As depicted in Figure 1 these influences on the technical system could arise from the environment, the user or the system itself. Every self-optimizing system features a system of objectives, whereas different objectives could be pursued by the system depending on the current situation and influences. The choice of one or more objectives leads to behavior adaptation. In general, there are two options for behavior adaptation: parameter or structural adaptation; whereas structural adaptation leads to compositional adaptation or reconfiguration. The determination of a suitable behavior is reflected by the self-optimization process, which is subdivided into three actions. In the first action the situation is analyzed. Therefore, the influences, the current system state and the current degree of performance concerning the system objectives are collected. From this the next objectives are deduced within the second action. In the third action the system adapts its behavior with respect to the chosen objectives.

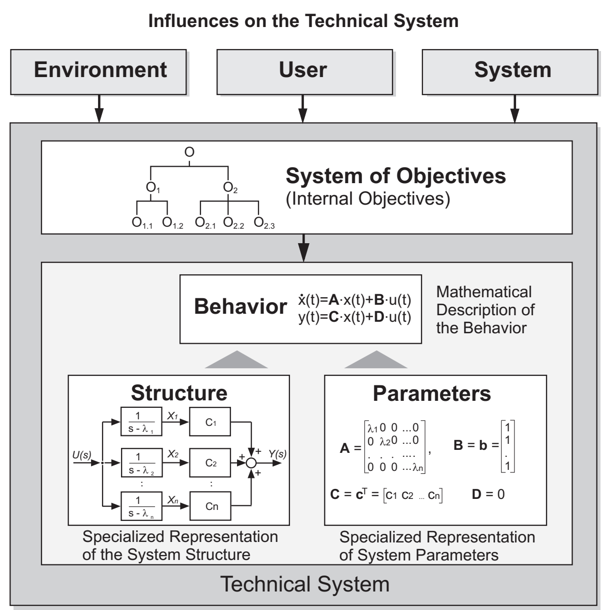


Figure 1. Aspects of self-optimization

As every mechatronic system self-optimizing systems comprise the physical system, sensors, actuators and the information processing. The information processing of self-optimizing systems is structured into three layers (see Figure 2) and is called operator-controller-module (OCM).

The self-optimization process takes place in the cognitive operator. The optimization itself can be model-based or behavior-based. Model-based techniques enable predictive, time-independent optimization. Behavior-based techniques provide planning and assessment functions with respect to the current objectives. The cognitive operator processes information in soft real time. On the contrary, the controller works in hard real time. The controller supervises the mechatronic system, reads sensor data and adjusts the actuators. The reflective operator is situated in between these two layers. It has the task to supervise the controller. The chosen behavior adaptation of the system is passed from the cognitive operator to the reflective operator, which has to initiate the correct control strategy as well as controller parameter variations. The configuration control, a finite state machine, is responsible for switching control strategies, but has no direct impact on the actuators. Further information about the operator-controller-module are given in [3].

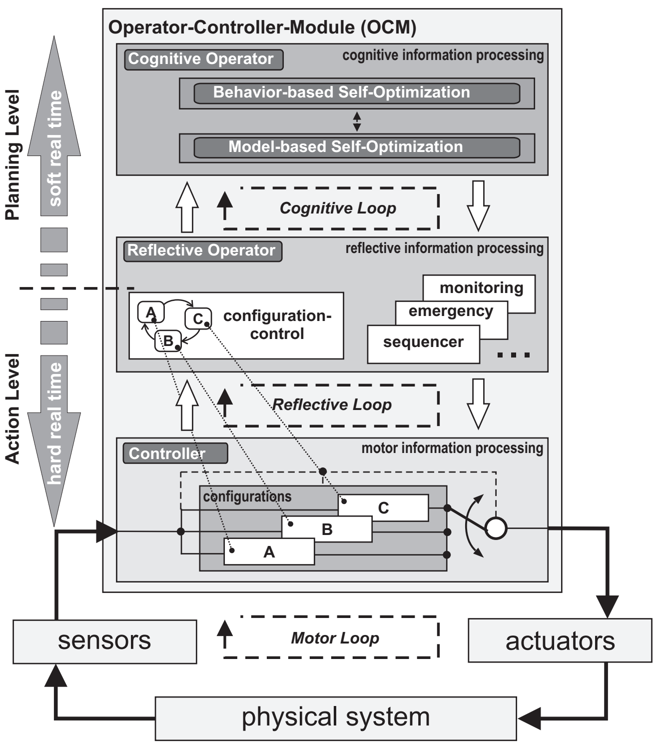


Figure 2. Operator-Controller-Module

Self-optimizing system are complex and due to their non-deterministic behavior comprise hidden risks, which cannot be foreseen in the design phase of the system. A condition monitoring policy is needed to supervise the system behavior and avoid severe dependability-critical impacts of self-optimization. Furthermore, self-optimization is able to increase dependability by integrating the objectives reliability, availability and safety in the system of objectives.

The paper is structured in five sections. This introduction to self-optimizing systems is followed by a short survey of related work regarding maintenance and condition monitoring in Section II. Section III introduces the proposed condition monitoring policy for self-optimizing systems. Section IV presents the active guidance module as an application for the developed policy. Afterward Section V illustrates results of first test drives, which should enhance the implementation of the proposed concept. Finally, Section VI presents a conclusion and a gaze at future work.

II. Related Work

According to the taxonomy proposed by Kothamasu et al. maintenance strategies can be subdivided into reactive and proactive maintenance, whereas the latter comprises preventive and predictive approaches [4]. Reactive maintenance is executed to correct a failure or avoid serious consequences, but not to avoid failures in advance. The preventive methods avoid failures, e.g., by fixed maintenance intervals, but only predictive maintenance monitors the current system behavior continuously. Due to the non-deterministic behavior of self-optimizing systems the latter strategy is preferable.

The two strategies, which are mentioned in the context of predictive maintenance, are reliability centered maintenance (RCM) and condition based maintenance (CBM). RCM is a combination of several maintenance strategies with respect to high reliability by minimum costs (cf. [5]). CBM is applicable, when the parameters, which should be monitored, are measurable by the system. Diverse techniques and methodologies for condition monitoring are well investigated [6] and applied for example in the field of advanced manufacturing [7]. The measured parameter is compared to an alarm value and if this value is exceeded the maintenance action is initiated. That is why Ma et al. [8] point out that condition monitoring in today's complex systems is mostly regarded as an alarm tool for maintenance. Even the ISO 17359 defines mainly, how a condition monitoring policy should look like to establish a successful maintenance strategy [9]. For a good review on CBM including diagnosis and prognostic techniques see, e.g., Jardine et al. [10].

None of these maintenance strategies influence the dependability of the system in the operating phase. Instead merely the next maintenance time is determined. Condition monitoring is used to assess the current state of the monitored system or component. The focus concentrates on the short term prediction. As consequence Ma et al. [8] call for a holistic view on the system and postulate that both, short and long term prediction, are required .

A first approach to calculate operational availability regarding optional machine configurations and technical services is given by Fleischer et al. [11]. Reliability and availability prognosis are established to optimize life cycle performance of the investigated machine tool.

In the operating phase newer publications describe self-healing, self-repairing and self-maintaining systems. As physical systems are not capable of repairing components like broken sensors, most of the concepts deal with software systems (cf. [12][13]). In the domain of mechanical systems Umeda [14] propose the development self-maintenance machines. The basis of these machines is functional redundancy as fault-tolerant scheme. As a first application a photocopier is designed self-maintaining [15].

III. Condition monitoring in self-optimizing systems

Self-optimizing systems comprise the risk of unforeseen failures due to their complexity and inherent non-deterministic behavior. Nevertheless self-optimization could be used to increase the dependability of the technical system. This section presents the established multi-level dependability concept and highlights how this concept could be embedded in an augmented condition monitoring policy.

A. Multi-level Dependability Concept

To reduce the risks and use the potentials of self-optimization the multi-level dependability concept (MLDC) has been developed (cf. [16]). This concept is situated in the reflective operator and subdivided into four hierarchically ordered levels (see Figure 3). On the one hand the concept offers well-directed measures to react on monitored circumstances and on the other hand the importance of the objective dependability is characterized and feed back to the cognitive operator.

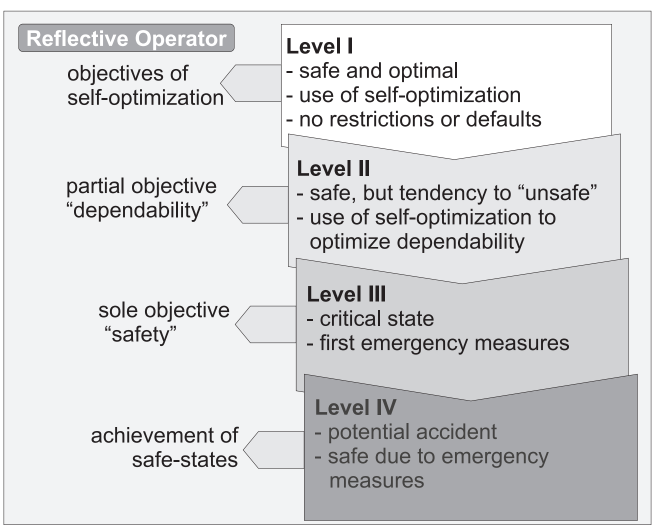


Figure 3. Multi-level dependability concept

The levels of the dependability concept and the impacts on the system of objectives are explained in detail in the following:

- Level I: The system operates in a dependable way. Dependability is one objective among others.
- Level II: An error has occurred. Self-optimization is used to return to the first level. Therefore, the priority of the affected attribute of dependability is increased.
- Level III: A severe error has occurred. First emergency mechanisms are triggered to reach a safer state. In the system of objectives safety is the sole objective to avoid the failure of the whole system and the consequences involved. The other attributes of dependability may occur as sub-objectives of safety.
- Level IV: The control over the system is lost. Mechanisms, like emergency routines, are executed to reach a fail-safe state.

In conjunction with the configuration control the multi-level dependability provides an interface between the fast controller structure and the self-optimization process in the cognitive operator.

B. Condition Monitoring Policy for Self-optimizing Systems

In this contribution the multi-level dependability is embedded as one part of the proposed policy of establishing condition monitoring. This policy is based on the common ISO 17359 policy [9]. This standardized policy has to be augmented and modified mainly in the operating phase by steps and details that utilize the advantages of self-optimization. In Figure 4 the steps of the adapted condition monitoring policy are illustrated. New or modified steps within the policy are highlighted in gray. The other steps (white background) are consistent with the ISO 17359. The policy is divided into three main phases: the design phase, the operating phase and the maintenance phase.

1) Design phase: The design phase of the condition monitoring policy should be parallel to the design phase of the system. The development of the system should be accompanied by the identification of self-optimizing potentials [17]. For self-optimizing systems the advantages of self-optimization over well-established design solutions must be exposed. This analysis highlights the variables and structures, which can be influenced by self-optimization and hence used for the condition monitoring proactive measures. Furthermore, the inherent objectives of the system are defined in this stage and obviously dependability should be one of them. Dependability in our case, as an abstract objective, is defined according to Laprie [18] by its four attributes reliability, availability, safety and security, whereas the latter plays a minor role in self-optimizing function modules.

In the subsequent reliability analysis several methods like reliability block diagram, failure mode and effects analysis, and fault tree analysis can be applied to classify the risks and to reveal dependencies and fault propagation paths. Being in parallel with the development process, an early reliability audit is able to avoid misconstructions and to add required system redundancy for critical functions. After the reliability analysis highlights parameters, which influence the dependability, these parameters should be mapped onto the dependability attributes safety, availability and reliability. The attributes are described by corresponding mathematical equations, e.g., failure rate or survival probability for reliability or mean time between failure (MTBF) for availability (cf. [19]). These steps make the function module comparable with other modules or the whole system.

The required parameters should be measurable to install proactive measures. As explained above, the preferable maintenance strategy for self-optimizing systems is predictive maintenance. As a consequence of this the proposed decision step of the ISO 17359 to install corrective or preventive maintenance is left out. Thus, in the current step suitable measurement methods for the predictive maintenance parameters are selected.

The next step in the proposed condition monitoring policy is the implementation of multi-level dependability concepts, which classify the current state on basis of the dependability quantities. In most cases there is one multi-level dependability concept for availability, safety and reliability. The implementation and hence the specification of the alarm values and the priority of the quantities depend on the investigated function module.

The next step is to set up the configuration control, which is also situated in the reflective operator. The configuration control choses the desired control strategy according to the current situation and is defined by a finite state machine. A state machine consists of states and transitions. The states arise from the different control concepts of the development phase. For example, redundancy and degradation concepts are reflected in this state machine. The transitions could be derived from the reliability analysis. A fault tree analysis for each state indicates the state transitions. The configuration control is a dependability-critical component and it has to be guaranteed that the state machine itself processes faultlessly. This can be done by model checking, which is extended for real time mechatronic systems [20][21].

2) Operating phase: The first step in the operation phase is taking measurements of the monitored parameters and calculate the resulting dependability quantities. This information is transmitted on the one hand to a hard real time information processing (left branch in Figure 4). The quantities are compared immediately with the multi-level dependability concept. If the state of the system is still Level I, the system is in a regular state. Otherwise the system performs known diagnosis and short term prediction methods to deduce the required action. This process step works in hard real time, which is necessary to initiate emergency routines. On the other hand the quantities are passed to the cognitive operator, which works in soft real time (right branch in Figure 4). On the basis of further knowledge the long term progression of the system is analyzed and the dependability quantities are updated for a point in time in the future. Afterward the retrieved quantities are compared with the multi-level dependability concept within the reflective

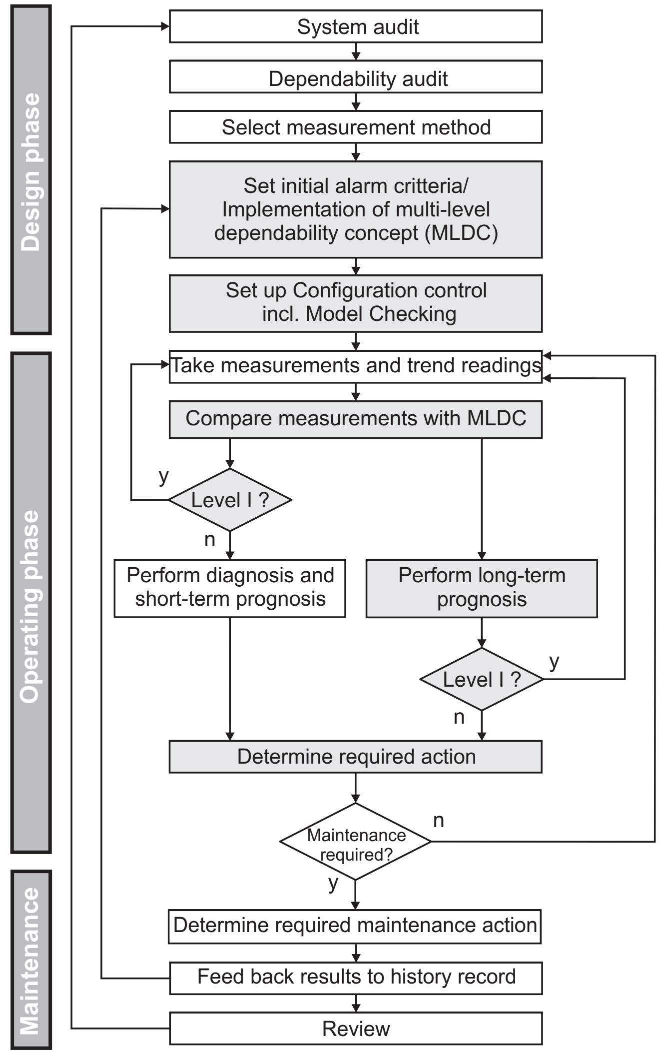


Figure 4. Condition Monitoring policy for self-optimizing function modules – new or modified steps are highlighted in gray

operator and checked if the system will still be in Level I for a finite horizon of time. If Level I is exceeded a suitable proactive measure has to be taken. An example for planning in combination with the multi-level dependability concept is given in [22].

In the subsequent step – if Level I is exceeded – the required action is determined. Basis of the decision is the current classification within the multi-level dependability concept. If the system state is in Level IV a fast reaction like an emergency routine is necessary. In Level II or Level III the objective dependability is increased in the system of objectives. Additionally, the configuration control can be triggered to change the control strategy. In a first approach the dependability Levels from the short term and long term prognosis are compared. The worse Level is used and fed back to the self-optimization process for weighting the objective dependability.

3) Maintenance phase: If maintenance is required, which should be the result of the short term and long term prediction in the previous step, the designated maintenance action is scheduled. As in the policy of the ISO 17359 the results are fed back to history record and prepared for review.

IV. Application to the active guidance module

The main demonstrator of the CRC at the University of Paderborn is an innovative railway system. Independent vehicles, called RailCabs, are able to transport passengers or goods non-stop from departure to destination. Moreover, these RailCabs form convoys to take advantage of slipstream and to reduce the energy consumption of the following RailCabs. To dissolve these convoys at switches at high velocities (intended maximum velocity is about 180km/h), the switch remain passive and the single RailCab steers in the desired direction. The responsible module for this steering action is called active guidance module. The module with its most important components is shown in Figure 5. Besides steering in passive switches, the guidance module reduces wear on wheels and rails by avoiding flange contacts on straight tracks as well as in curves by compensating disturbances like track irregularities and side wind. Other influences on the active guidance module could occur from the RailCab itself, which, for example, restricts the energy consumption of the module or varies the velocity. During the operating phase the human has currently no influence on the system of objectives and therefore is not considered.

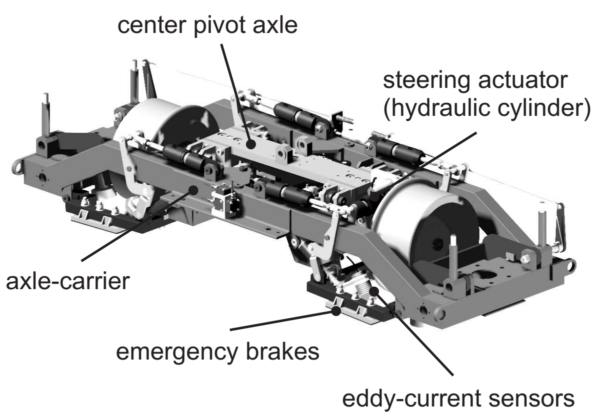


Figure 5. Active guidance module

In the following the proposed condition monitoring policy (Figure 4) is applied to this active guidance module. Within the system audit the potential of self-optimization should be pointed out. The controlled parameter in the active guidance module is the steering angle. This angle is calculated by an optimized trajectory, which utilizes the track clearance to adapt to the desired objective. In general, there are two antagonistic objectives that the active guidance module can pursue reasonably.

On one hand the objective is to reduce energy consumption. This leads to the reduction of steering activity and trajectories that somehow “cut curves”. This results in a higher probability of flange contacts and a reduction of the availability of the system due to wear on wheels. Hence, on the other hand the objective is to minimize the wear on wheels, which leads to a trajectory near the center line of the track clearance to avoid flange contacts. This leads to a conflict within the system of objectives and thus to a multi-objective optimization which result is a Pareto set for the chosen objectives. More information about the generation of optimized trajectories using model-based optimization within the guidance module can be found in [23].

In the dependability audit a first failure mode and effects analysis (FMEA) is conducted to find non-tolerable risk and measures like additional redundancy to secure the system. This analysis is based on the principle solution of the guidance module. One result of the FMEA is that the determination of the lateral position, which is measured by eddy currents sensors, is important for the control of the guidance module as failures in the measurement can lead to severe faults like the derailment of the RailCab. Therefore, the eddy current sensors are attached in active redundancy. Two pairs of sensors for each guidance module are installed, whereas one pair requires that both sensors are working. If both pairs of sensors are working well, the mean is used to enhance the accuracy of the sensor signal. If one pair of sensor fails, this leads to a deterioration of the signal. For measuring the longitudinal position four incremental sensors, each at a wheel, are used. The mean is calculated to refine the measurement. As the incremental sensors are subjected to drift, the sensors are updated by proximity switches every 15m.

The next step is to link the physical parameters to the dependability attributes. Wear on wheels, for example, is inversely proportional to the availability of the guidance module. The accurate relation has to be determined experimentally, whereas as a matter of fact flange contacts lead to higher wear. Considering the dependability attributes safety the physical quantity to keep the system safe is the lateral acceleration. The lateral acceleration should be low to avoid derailment by hard flange contact. Regarding reliability, the state of each component required for the controlling actions is monitored. Sensor failures could have an serious impact on the reliability of control strategies.

A fault-tree analysis (FTA) is used to determine which events have to occur to force system degradation. Additionally, the quantitative FTA provides an indication of setting the alarm values of the multi-level dependability concepts regarding reliability. To set the alarm levels of the attribute availability and investigate which parameter would support undesired flange contacts, several test drives are made. Figure 6 shows the different control strategies, which are also specified in the design phase, and the classification considering the multi-level dependability concept with respect to reliability. The control strategies include in most cases a track position controller, which gets a desired trajectory as set value, and a feed forward control based on the current curvature of the track. In Level I the trajectory is generated by the self-optimization process with a high priority on the minimization of the energy consumption. If a failure in an eddy current sensor occurs or the self-optimization process produces inadequate results the system state is classified into Level II. In the first case the self-optimization process assigns more priority on the dependability of the system the reliability and availability of the system increases. In the latter case the center line of the clearance is used as fallback trajectory for the track position controller (track position controller A - both sensors pairs working). If one pair of sensors fails without having the possibility to use self-optimization the state is sorted into Level III (track position controller B). The map-based feed forward control is used, if all eddy current sensors fail. In this control strategy the steering angle is only set due to the curvature of the track. The emergency routine in the fourth level is fixing the axle to avoid uncontrolled behavior. So only in the first two levels self-optimization is used to enhance the dependability.

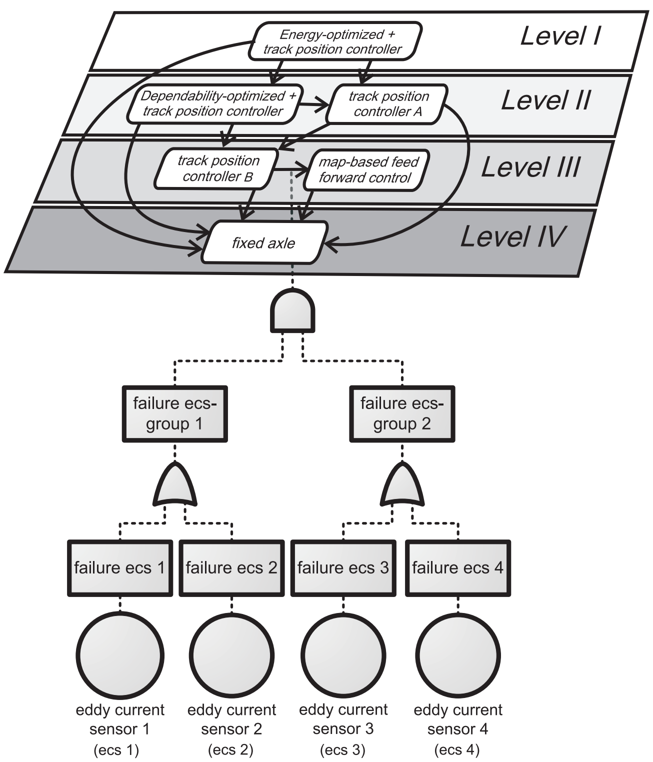


Figure 6. Interrelation Configuration Control and FTA

Figure 6 illustrates that these different control strategies are utilized as states in the state chart of the configuration control. As the states are derived from the given control structure the transitions could be derived from the FTA by choosing the failure of every control strategy as top event. For the determination of the transitions every fault tree is divided into minimal cut sets. The exact policy is descibed in DIN EN 61025 [24]. At the moment, expert knowledge is used to determine which minimal cut set belongs to which transition. One example of such a cut set is shown in Figure 6, which is linked to the transition manually. The condition of the transition could be derived by the Boolean Algebra, which is given by the fault tree notation. As explained above, the four redundant eddy current sensor (ecs) are ordered in the following way. Two sensors work together as a pair. If one sensor fails then the whole pair fails. So this relation is indicated by an OR-Gate. The other pair is still working. If also the second pair fails the track position controller is not able to work anymore (AND-Gate) and the transition of the state chart is triggered and the state switches to the map-based feed forward control. The adequate Boolean expression, which is deduced from the fault tree is

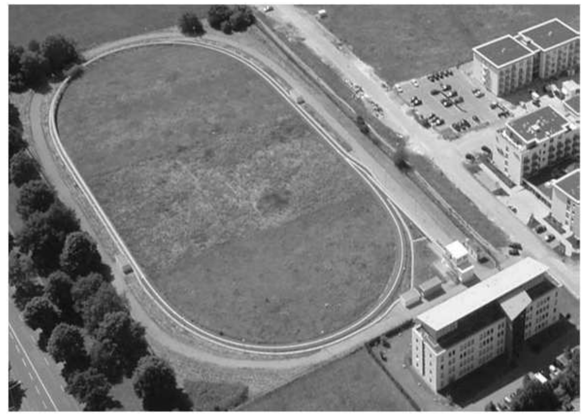
$$(ecs1 \vee ecs2) \wedge (ecs3 \vee ecs4). \tag{1}$$

Regarding the guidance module the attributes reliability and availability are investigated. The reliability is reduced, if, e.g., certain failures in the hardware occur. The paper focuses on sensor failures, which handicap the control of the guidance module. Availability is quite similar as it is assumed that the number of flange contacts rises if a failure occurs. So the number of flange contacts could be used to assess the current availability of the system, respectively, to calculate the residual life time. As mentioned above, the established state chart refers to the fault tree analysis of the different control strategies. The ordering of the different control strategies to a certain alarm level is based on expert knowledge. It is depicted that only degradation processes are possible. The repair of the system takes place in the maintenance phase of the system. To avoid undesired behavior of the system, an advanced model checking is conducted on the state chart (cf. [25]). The model checking has to guarantee, for example, that the fail-safe state could be reached from every current state if a severe failure occurs. This concept is implemented on the active guidance module. To validate and enhance this concept it is necessary to take a closer look on the operating phase of the system. Therefore, several test drives were made to examine which parameter has got a relevant influence on the system behavior. In the next section selected results of the test drives are shown.

V. Experimental results

The test drives were conducted on the test track next to the University of Paderborn. In Figure 7 the test bed is shown. On the left side the RailCab is depicted. The RailCab on a scale of 1:2.5 is able to drive on the test track with a maximum velocity of about 10 m/s. The oval test track

(a) RailCab on a scale of 1:2.5



(b) Test track next to the University of Paderborn

Figure 7. Test bed

is 450m in length. The described multi-level dependability concept and the configuration control are programmed with Matlab/ Simulink and transferred to the dSPACE real time hardware on the RailCab.

In the following three sections the results of different test drives are illustrated, whereas each section has a different focus. In the first section the control strategies are investigated. Besides driving with the track position controller in energy-optimized and dependability-optimized mode failures are induced to gain knowledge about the behavior in the case of a failure. For a comparison also the map-based feed forward control is investigated. In the second section the influence of different track sections is examined. The test track is subdivided into straights, clothoids and curves. The third section shows results for driving at different velocities.

The assessment parameter of all test drives is the summed length of the flange contacts q_{fc}. The flange contacts are calculated by the relation between the clearance c and the lateral position r_y

$$q_{fc} = \frac{2r_y}{c}. \tag{2}$$

If $q_{fc} \geq 1$ a flange contact is detected. The length of the flange contact is calculate by the determination of the beginning and end of each flange contact.

A. Comparison of different control strategies

The failure mode and effects analysis within the design phase of the proposed policy points out that the eddy current sensors are dependability-critical components. As consequence they are built up in hardware redundancy. This offers the option to use the mean of both sensor pairs to get an more accurate signal. But if one pair fails this leads to a reduction of the signal quality. If both pairs fail, the map-based feed forward control is the last possible control strategy. This section shows a comparison between the different control strategies and the impact of failures of the eddy current sensors. Table I lists the different test drives, which are investigated. All test drives were done at a velocity of 5m/s. The track sections of the passive switch are neglected in the evaluation because in the passive switch the active guidance module steers the RailCab next to the left or right rail to pass the switch. So a lot of flange contacts are expected, which would bias the results.

Table I
TEST DRIVES WITH DIFFERENT CONTROL STRATEGIES

Test drive	Control strategy	failure of eddy current sensors
1	Dependability-optimized + track position controller	no failure
2	Energy-optimized + track position controller	no failure
3	Energy-optimized + track position controller	failure one pair
4	Dependability-optimized + track position controller	failure one pair
5	map-based feed forward control	failure both pairs

The results of the first test drives are depicted in Figure 8. The flange contacts are shown in m. The complete track without the passive switch sections is about 350m in length. Comparing the two rounds it is obvious that the test drives are highly reproducible. The comparison of drive number one and two shows that the dependability-optimized trajectory leads to lesser flange contacts in m than the energy-optimized trajectory. This is expected as the dependability-optimized trajectory tries to keep the RailCab next to the center line of the clearance, whereas the energy-optimized trajectory gets quite close to the rails to reduce the energy consumption. This could result in flange contacts due to the underlying controller of the hydraulic actuator. Furthermore, the figure illustrates that the flange contacts in m increase if one pair of eddy current sensor fails (drive number three and four compared to drive number one and two). For the availability of the system it is an option to change to the dependability-optimized trajectory in the case of a failure. The fifth test drive shows that the summed length of flange contacts increase significantly if the guidance module has to switch to the map-based feed forward control.

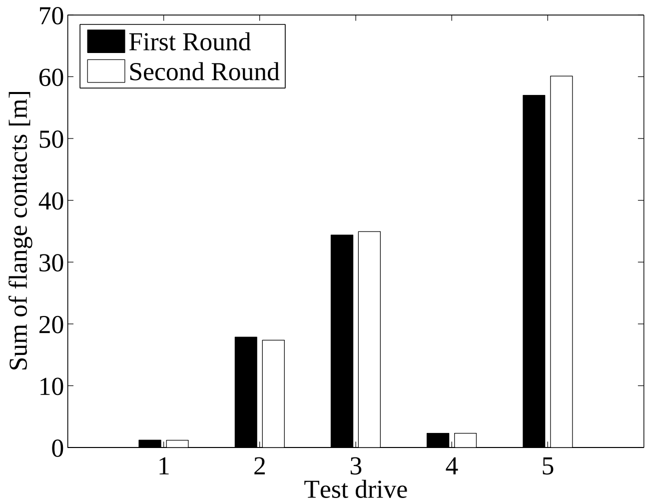


Figure 8. Flange contact in different control strategies

As these results show, self-optimization, e.g., different objectives like energy-optimized or dependability-optimized, leads to significant different amounts of flange contacts. Furthermore, it is illustrated, that in the case of a sensor failure the amount of flange contacts increases in both modes. But a change in the objectives from energy-optimized to dependability-optimized could decrease the number of flange contacts and thus the wear on the wheels. For the comparison the objectives have either high priority on energy or high priority on dependability. A compromise of both objectives as calculated within the multi-objective optimization could be more appropriate.

B. Comparison of track sections

In the preceding section is discussed, that sensor failures can lead to higher flange contacts. A higher number of flange contacts can lead to an adaptation of the behavior. For the condition monitoring it plays a decisive role which factors or influences can also lead to an increased amount of flange contacts. In this and the following section two influences are discussed: the influence of the current track section and the influence of the velocity of the RailCab. If the impact of these influences is known we are able to use it for the analysis of the current situation. Considering long-term prognosis we are able to adapt the behavior in advance.

The test track consists of three different track sections: straights, clothoids and curves. Figure 9 shows the results for driving two rounds with 5m/s on the test track. The first three bars refer to drives with the dependability-optimized control strategy (two of them cannot be seen due to the flange contacts are zero) and the second three bars refer to drives with the energy-optimized control strategy. It is illustrated by comparing both strategies, that in every track section type the dependability-optimized strategy counts lesser flange contacts (cf. preceding section). The comparison of the track section is given in percent. This is due to the different length of the track sections. The length of the straight and clothoid is 50m, whereas the length of the curve is 77m. Within the dependability-optimized drives only in curves were some flange contacts (0.18%). It is depicted that driving a straight road in the energy-optimized way leads to lesser flange contacts that in the clothoid and driving a clothoid produces lesser flange contacts than driving a curve. This is due to the lateral accelerations, which forces the RailCab to drive more on the outer rail. The flange contacts increase from straight to curved track because the trajectory has to compensate the curvature in addition to the track irregularities.

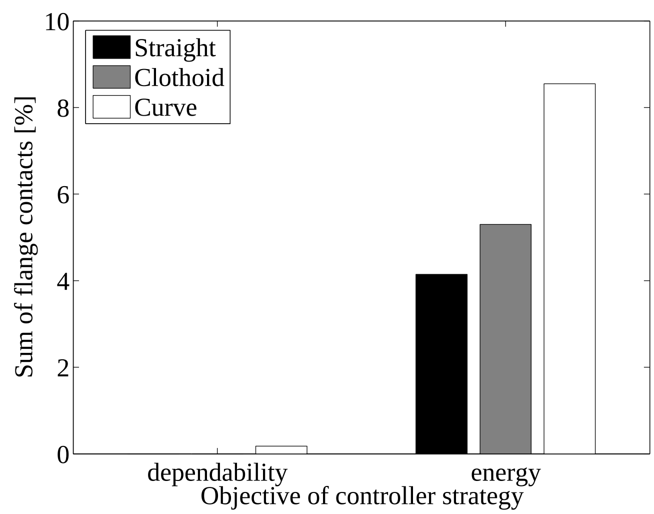


Figure 9. Comparison of different track sections

The showed effect leads to several consequences. First of all it is obvious that the current track type has to be considered while analyzing the current situation. Furthermore, the track type could be one parameter regarding the long-term prognosis of the guidance module. If it is desirable to increase the availability by reducing the wear on wheels and rails, it is preferable to assign a higher priority on the objective dependability in curves.

C. Comparison of different velocities

The comparison of different velocities is important, because a measure to react on failures and thus to lower the amount of flange contacts could be the reduction of the velocity of the RailCab. Figure 10 illustrates the results of the test drives regarding the velocity. Again, the first three bars refer to the dependability-optimized and the second three bars refer to the energy-optimized control strategy. The test drives were executed at 3m/s, 5m/s and 7m/s.

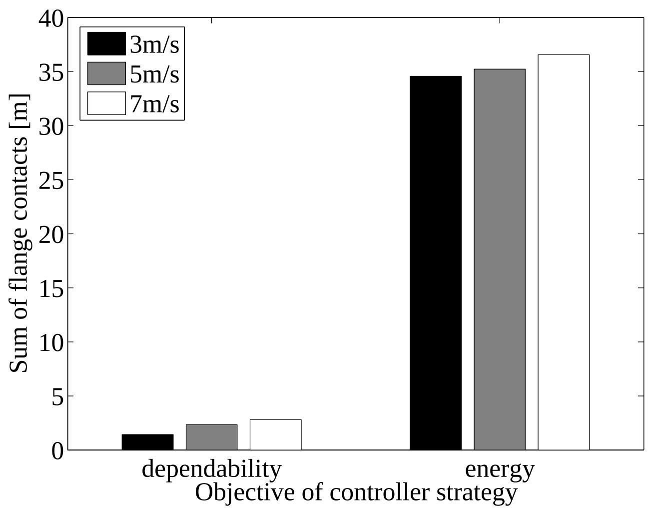


Figure 10. Comparison of different velocities

As it can be seen in Table II the results are highly reproducible. The change in flange contacts regarding the velocity is low. Only in the first round driving with 3m/s the resulting flange contacts (17.954m) are higher than expected. Overall it could be stated that the higher the velocity the higher is the amount of flange contacts in m. The difference between the optimization control strategies is again significant. Driving in the dependability-optimized control strategy always leads to lesser flange contacts.

Table II
FLANGE CONTACTS [M] AT DIFFERENT VELOCITIES

control strategy	round	3m/s	5m/s	7m/s
Dependability-optimized + track position controller	1	0.953	1.184	1.249
Dependability-optimized + track position controller	2	0.474	1.159	1.558
Energy-optimized + track position controller	1	17.954	17.869	18.042
Energy-optimized + track position controller	2	16.612	17.354	18.507

In the end, reducing the velocity could be a mean to reduce flange contacts. In combination with a change in the control strategy from energy-optimized to dependability-optimized the flange contacts could be lowered even more. In our test cases the effect of reducing the velocity is small. Even smaller than setting the priority of the optimization more to dependability strategy. This difference will be more significant at higher velocities.

VI. CONCLUSION AND FUTURE WORK

This contribution shows a policy for increasing the dependability of self-optimizing systems. The ISO 17359 condition monitoring process is modified to combine condition monitoring and self-optimization. In today's applications condition monitoring is seen as a tool for the succeeding maintenance action, whereas the policy proposed in this contribution focuses on the operating phase. The multi-level dependability concept is embedded in this policy to avoid risks from self-optimization and to increase the objectives availability, reliability and safety. So systems with a self-optimizing component constitute a new kind of redundancy to reach the highest reliability or availability level that is feasible. The proposed policy is applied to the active self-optimizing guidance module of an innovative rail-bound vehicle.

On one hand the experimental results illustrate that self-optimization is able to compensate sensor failures by changing the objective from "minimizing energy consumption" to "maximize dependability". Additionally, the difference in driving the established control strategies considering the flange contact are pointed out. For the comparison of test drives the test drives were conducted with a high priority on dependability or a high priority on reducing energy consumption. It is preferable to use a smooth or continuous modification of the priorities in the future. On the other hand the results give first advices for the integration of prognosis. The current track section and the current velocity should be considered. The gained information could also be use to enhance the long-term prognosis of the residual life time of the guidance module. So the next step in testing will be the use of diagnostics as well as short and long term prediction. The combination of both is already destined in the proposed condition monitoring policy. As the long term prediction is also done within the cognitive operator by the planning component this is an interesting point of research for the integration of condition monitoring in self-optimizing systems.

At the current point of research the work is explicit for mechatronic function modules like the active railway guidance module. A further step is to interlink the diverse function modules of the RailCab and examine the whole system with respect to reliability-centered condition monitoring.

ACKNOWLEDGMENT

This contribution was developed in the course of the Collaborative Research Center 614 "Self-Optimizing Concepts and Structures in Mechanical Engineering" funded by the German Research Foundation (DFG). The authors would also like to thank Jens Geisler who accompanied the test drives.

REFERENCES

[1] C. Sondermann-Wölke and W. Sextro, “Towards the integration of condition monitoring in self-optimizing function modules,” in *2009 Computation World: Future Computing, Service Computation, Cognitive, Adaptive, Content, Patterns, Nov. 15-20, 2009, Athens, Greece*, 2009, pp. 15–20.

[2] J. Gausemeier, S. Kahl, and S. Pook, “From mechatronics to self-optimizing systems,” in *Self-optimizing Mechatronic Systems: Design the Future, 7th International Heinz Nixdorf Symposium, Feb. 20-21, 2008, Paderborn, Germany*, 2008, pp. 3–32.

[3] T. Hestermeyer, O. Oberschelp, and H. Giese, “Structured information processing for self-optimizing mechatronic systems,” in *Proc. of 1st International Conference on Informatics in Control, Automation and Robotics (ICINCO 2004), Setubal, Portugal*, 2004, pp. 230–237.

[4] R. Kothamasu, S. Huang, and W. VerDuin, “System health monitoring and prognostics–a review of current paradigms and practices,” *The International Journal of Advanced Manufacturing Technology*, vol. 28, no. 9, pp. 1012–1024, 2006.

[5] M. Rausand, “Reliability centered maintenance,” *Reliability Engineering & System Safety*, vol. 60, no. 2, pp. 121–132, 1998.

[6] A. Davies, Ed., *Handbook of Condition Monitoring – Techniques and Methodology*. Chapman and Hall, 1998.

[7] L. Wang and R. X. Gao, Eds., *Condition Monitoring and Control for Intelligent Manufacturing*, ser. Advanced Manufacturing. Springer London, 2006.

[8] L. Ma, “Condition monitoring in engineering asset management,” in *12th Asia-Pacific Vibration Conference, August 6-9, 2007, Sapporo, Japan*, 2007.

[9] DIN ISO 17359, “Condition monitoring and diagnostics of machines – general guidlines,” 2003.

[10] A. Jardine, D. Lin, and D. Banjevic, “A review on machinery diagnostics and prognostics implementing condition-based maintenance,” *Mechanical Systems and Signal Processing*, vol. 20, no. 7, pp. 1483–1510, 2006.

[11] J. Fleischer, S. Niggeschmidt, and M. Wawerla, “Optimizing the Life-Cycle-Performance of Machine Tools by Reliability and Availability Prognosis,” in *Advances in Life Cycle Engineering for Sustainable Manufacturing Businesses: Proceedings of the 14th Cirp Conference on Life Cycle Engineering, Waseda University, Tokyo, Japan, June 11th-13th, 2007*. Springer, 2007, p. 329.

[12] D. Garlan, J. Kramer, and A. L. Wolf, *Proceedings of the First Workshop on Self-Healing Systems, WOSS 2002, Charleston, South Carolina, USA, November 18-19, 2002*. ACM, 2002.

[13] S. Takata, F. Kirnura, F. Van Houten, E. Westkamper, M. Shpitalni, D. Ceglarek, and J. Lee, “Maintenance: changing role in life cycle management,” *CIRP Annals-Manufacturing Technology*, vol. 53, no. 2, pp. 643–655, 2004.

[14] Y. Umeda, T. Tomiyama, H. Yoshikawa, and Y. Shimomura, “Using functional maintenance to improve fault tolerance,” *IEEE Expert*, vol. 9, no. 3, pp. 25–31, 1994.

[15] Y. Shimomura, S. Tanigawa, Y. Umeda, and T. Tomiyama, “Development of self-maintenance photocopiers,” *AI Magazine*, vol. 16, no. 4, pp. 41–53, 1995.

[16] C. Sondermann-Wölke, J. Geisler, T. Müller, A. Trächtler, and J. Böcker, “The active guidance module of a rail-bound vehicle as an application for the dependability oriented design in self-optimizing systems,” in *ASME2008 - International Design Engineering Technical Conferences & Computers and Information in Engineering Conference, New York*, 2008.

[17] J. Gausemeier, U. Frank, and D. Steffen, *Reconfigurable Manufacturing Systems and Transformable Factories*. Springer, 2006, ch. 36: Intelligent Systems, Self-optimizing Concepts and Structures, pp. 719–742.

[18] J.C. Laprie, Ed., *Dependability: Basic Concepts and Terminology in English, French, German, Italian, and Japanese*. Wien: Springer, 1992.

[19] A. Birolini, *Reliability Engineering – Theory and Practice*, 3rd ed. Springer Berlin, 2003.

[20] H. Giese, S. Burmester, W. Schafer, and O. Oberschelp, “Modular design and verification of component-based mechatronic systems with online-reconfiguration,” *ACM SIGSOFT Software Engineering Notes*, vol. 29, no. 6, pp. 179–188, 2004.

[21] H. Giese and M. Hirsch, “Modular Verification of Safe Online-Reconfiguration for Proactive Components in Mechatronic UML,” *Lecture Notes in Computer Science*, vol. 3844, pp. 67–78, 2006.

[22] B. Klöpper, C. Sondermann-Wölke, C. Romaus, and H. Vöcking, “Probabilistic planning integrated in a multilevel dependability concept for mechatronic systems,” in *2009 IEEE Symposium on Computational Intelligence in Control and Automation, Mar. 30 - Apr. 2, 2009*. IEEE, 2009, pp. 104–111.

[23] J. Geisler, K. Witting, A. Trächtler, and M. Dellnitz, “Multiobjective optimization of control trajectories for the guidance of a rail-bound vehicle,” in *Proceedings of the 17th World congress The International Federation of Automatic Control*, 2008, pp. 4380–4386.

[24] IEC 61025:2006, “Fault tree analysis (FTA); German version EN 61025:2007,” August 2007.

[25] C. Sondermann-Wölke, J. Geisler, M. Hirsch, and T. Hemsel, “Verlässlichkeit im aktiven selbstoptimierenden Spurführungsmodul eines schienengebundenen Fahrzeugs,” in *6. Paderborner Workshop Entwurf mechatronischer Systeme, Apr. 2-3, 2009*, 2009, pp. 231–243.

Self-Organization based Network Architecture and Control Technologies for New Generation Networks

Naoki Wakamiya, Shin'ichi Arakawa, and Masayuki Murata
Graduate School of Information Science and Technology, Osaka University
Suita, Osaka 565-0871, Japan
Email: {wakamiya, arakawa, murata}@ist.osaka-u.ac.jp

***Abstract*—An emerging new generation network is requested to accommodate an enormous numbers of nodes with high diversity and a wide variety of traffic and applications. To achieve higher scalability, adaptability, and robustness than ever before, in this paper we present new network architecture composed of self-organizing entities. The architecture consists of the physical network layer, service overlay network layer, and common network layer mediating them. All network entities, i.e., nodes and networks, behave in a self-organizing manner, where the global behavior emerges through their operation on local information and direct and/or indirect mutual interaction intra- and inter-layers. We show several results to demonstrate how self-organizing network control behaves based on our architecture.**

***Keywords*-new generation network; network architecture; self-organization; bio-inspired algorithms**

I. Introduction

To satisfy a wide range of requirements and desire of people and to support our daily life in many aspects, a variety of fixed devices such as PCs, servers, home electric appliances, and information kiosk terminals, mobile devices such as that equipped with people and vehicles, and small and scattered devices such as RFID tags and sensors, are and will be distributed in the environment. They are and will be connected with each other and organize networks. They cooperate with each other in sharing and exchanging obtained or generated information and controlling each other to provide people with desired services for safe and comfortable environment.

Those devices generate a great variety of traffic including voice, video, computer, sensing, identification, control, and management data in accordance with a type of device, application, service, and context. Traffic characteristics also have the diversity, e.g., constant/intermittent, low/high rate, and small/large amount. Furthermore, the number, type, location, and usage of devices, condition of communication environment, and traffic characteristics dynamically and considerably change every moment.

It means that new generation networks is requested to accommodate an enormous numbers of heterogeneous devices and a wide variety of traffic with substantial fluctuation under dynamically changing communication environment [1]. Therefore, a network would often face unexpected or unpredictable user behavior, usage of network, and traffic pattern, which are beyond the scope of the assumption in designing and building the network. As a result, the performance considerably deteriorates or at worst the network completely collapses. Consequently, the conventional network design methodology, where structures, functionalities, algorithms, and control parameters are optimized to accomplish the best performance assuming certain operating environment, and fault detection, avoidance, and recovery mechanisms are prepared and preprogrammed for expected failure, is no longer feasible.

Taking into account requirements for a new generation network stated above, in [2] we propose new network architecture, which is more scalable to the number of nodes and scale of network, more adaptive to a wide variety of traffic patterns and their dynamic change, and more robust to expected and unexpected failure independently of size and duration, than ever before. Our basic idea is to organize and control the whole network system in a self-organizing manner where the global behavior emerges from mutual interaction among localized behavior of small entities. A network has a layered architecture; the physical network layer, the service overlay network layer, and the common layer, which mediates inter and intra layer interaction. Behavior of all entities constituting a network system, i.e., node, network, and layer, is self-organized. A node performs MAC, scheduling, routing, congestion control, and other control by using nonlinear functional modules called *self-organization engines*, which operate based on local information obtained through observation of environment and information exchange with neighboring nodes. Nodes further organize and control a network through localized behavior and mutual interaction among them. Networks within a layer also behave in a self-organizing way and interact with each other directly by exchanging messages and/or indirectly by changing operating environment shared among them. In addition to the intra-layer interaction, service overlay networks and physical networks interact with each other through mediation of the common network layer.

In the following sections, we introduce the self-organizing network architecture first starting with basic concept and

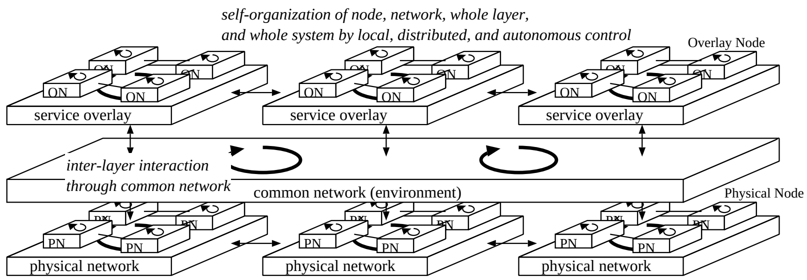


Figure 1. Self-organizing network architecture

followed by node architecture and components. We also show examples of combination of multiple self-organization engines and hierarchical control of network by a single self-organization engine. Then, we conclude the paper by mentioning related work and future issues.

II. Self-organizing network architecture

In this section, we briefly describe our network architecture together with some sample biological models for self-organized control of new generation networks.

A. Basic Concept

As the number of nodes and the size of network increase, a centralized mechanism becomes ineffective for considerable maintenance overhead to collect up-to-date information on the whole network system and distribute the decision to all nodes. Especially when we consider wireless communication, such control overhead occupies the limited bandwidth and disturbes regular data communication. Even semi-distributed mechanisms, such as a table-driven routing protocol, where nodes in a network perform a distributed control algorithm, requires nodes to keep the same and consistent up-to-date view of network. Therefore, we need fully distributed and autonomous control mechanisms, which enable a node to operate without the need for global information, but purely on local information obtained through observation of its surroundings and messages exchanges with neighbors. With such autonomous mechanisms, it also is possible to avoid letting a single and local failure, e.g., link disconnection, involve the whole system by propagating the failure information to update the topology information that all nodes maintain.

In addition, a conventional adaptation mechanism where the whole system is periodically re-optimized based on the up-to-date status information puts too much burden on a large-scale network to adapt to frequent changes in the operating environment. As the temporal order of changes in network condition becomes small in new generation networks, frequency of information update increases and becomes more harmful. Therefore, we need self-adaptive and self-configuration control mechanisms, which are local and distributed. Each node should autonomously and locally adapt control parameters, behavior, and even algorithm and mechanism in accordance with the state of surrounding environment.

Furthermore, a conventional network system acquires the robustness by implementing a variety of detection, avoidance, and recovery mechanisms against failure, error, abuse, extreme operating condition, and critical event. Such design methodology makes a network system complicated, monolithic, and even fragile. Therefore, we need simple and module-based control mechanisms where a node, network, and network system are constituted by autonomous, simple, and interacting functional control modules. When a part of the modules halts for unexpected failure, the remaining modules provide the minimum level of network service and provoke adaptive behavior of other modules and entities. Consequently, the whole network system adapts to the new environment.

In summary, we need a self-organizing network system where a node consists of autonomous and simple control mechanisms, mutual and local interaction among nodes organizes a network, and inter and intra layer interaction among networks organizes the whole network system. Such a network system can keep providing network services to users and applications independently of the size of system and condition of operating environment, the degree of their diversity and dynamic change, and the scale and duration of failure. The self-organizing network architecture we propose has three layers. They are the physical network layer consisting of wireless and wired access networks and optical core networks, the service overlay network layer consisting of service or application-oriented overlay networks, and the common network layer, which mediates interaction among entities within a layer and interaction among the two layers. These layers are self-organized through inter and intra-layer mutual interaction among entities. The architecture is illustrated in Figure 1.

B. Node Architecture

In the self-organizing network architecture, each of physical and overlay nodes is composed of communication and sensing module, knowledge database module, and self-organization engines, and network control functionalities

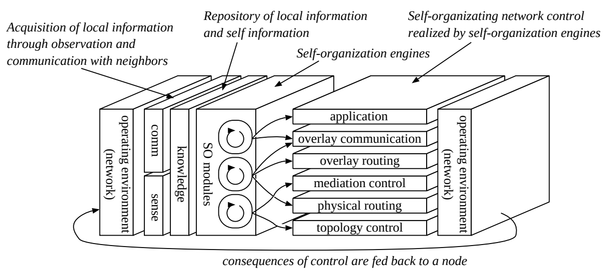


Figure 2. Node architecture

(see Figure 2). The communication and sensing module obtains local information through message exchange with neighboring nodes and observation of environmental condition by probing or sensing for example. The module also collects status information of node itself. Obtained information is deposited into the knowledge database to be used by self-organization engines.

A self-organization engine is a basic component for self-organizing behavior of node. It operates on local information in the knowledge database and reacts to its dynamic change. By using self-organization engines, a node realizes and performs MAC, scheduling, routing, congestion control, and other network control functionalities.

Behavior of a node changes the operating environment and affects neighbors. For example, emission of a message consumes the wireless bandwidth and may cause collisions and congestion. Neighbors would react to the change and behave accordingly. Such changes in the operating environment and neighbors' behavior are observed by the communication and sensing module and fed back to the node itself.

C. Self-Organization Engines

A self-organization engine is a nonlinear functional module and a core of self-organization. It operates on a nonlinear mathematical model in the form of differential equation. Examples of nonlinear models include a pulse-coupled oscillator model [3], a reaction-diffusion model [4], a response threshold model [5], and an attractor selection model [6]. All of these models are derived from self-organizing behavior of biological systems, which are inherently fully-distributed, autonomous, and self-organizing. As a typical example, it is well known that a group of social insects such as ants, termites, and honey bees often exhibits sophisticated and organized behavior, e.g., ant trail, cemetery formation, brood sorting, and division of labor, which is beyond mere collection of simple behavior of individuals. Such collective intelligence, called *swarm intelligence*, emerges from mutual and local interaction among simple agents [7]. By adopting such bio-inspired mathematical models to network control, we can expect to achieve a robust, scalable, and adaptive self-organizing network system.

In this subsection, we first introduce the above mentioned four nonlinear functions and how they are applied to specific network control. Then we give two examples of extended model, i.e., a combination of two nonlinear modules and a layered nonlinear module. We should note here that not all modules are incorporated to the proposed architecture and presented results are obtained by independent implementation and experiments.

1) Pulse-coupled Oscillator Model: A pulse-coupled oscillator model explains synchronized behavior observed in a group of flashing fireflies [3]. A firefly periodically flashes based on its biological timer keeping its intrinsic frequency when it is alone. When fireflies form a group, a flash of firefly stimulates non-flashing fireflies. A stimulated firefly advances its timer by a small amount. When the advanced timer reaches a certain threshold, the stimulated firefly flashes as well. At this moment, these fireflies are considered synchronized. By repeatedly stimulating each other through flashes, a group of fireflies eventually get synchronized and they begin to flash at the same time and at the same frequency.

In a pulse-coupled oscillator model, an oscillator maintains a timer. It fires when the phase of timer ϕ reaches one and then the phase goes back to zero. The dynamics of phase ϕ is formulated as,

$$\frac{d\phi_i}{dt} = \frac{1}{T_i} + \frac{\Delta(\phi_i)}{|\{j|j \in \mathcal{N}_i, \phi_j = 1\}|} \sum_{j \in \mathcal{N}_i} \delta(1 - \phi_j). \quad (1)$$

In (1), T_i stands for the intrinsic interval of oscillator i's timer. $\mathcal{N}_i$ is a set of oscillators coupled with oscillator i. Oscilalltors coupled with a firing oscillator are stimulated. $\Delta(\phi_i)$ is a monotonically increasing nonlinear function, which determines the amount of stimulus when oscillator i receives a fire. The global synchronization, where all oscillators flash simultaneously at the same frequency, can be accomplished without all-to-all coupling. As far as the network of oscillators is connected, oscillators with similar intrinsic interval are eventually synchronized. Depending on parameters and functions, so-called phase-lock condition, where oscillators fire alternately keeping the constant phase difference, can also be accomplished and a traveling wave appears.

A direct application of the pulse-coupled oscillator model is synchronization or scheduling. By regarding a wireless sensor node as a firefly and radio signal transmission as flash of a firefly, we can self-organize synchronization in a wireless sensor network. In Figure 3, phase transition in a network of 100 nodes randomly distributed in the region of 100 m$\times$100 m is shown. The communication range is identical among nodes and set at 20 m. Intrinsic frequency $F_i = 1/T_i$ is chosen within the range of [0.9,1.1] at random considering timer drift and error. Initial phase ϕ_i is also chosen at random taking into account error and asynchronous power activation. The x-axis of the figure corresponds to time and the y-axis shows state transition. Initially, phases

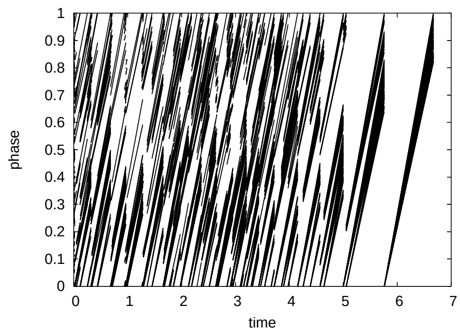


Figure 3. Synchronization in a wireless sensor network

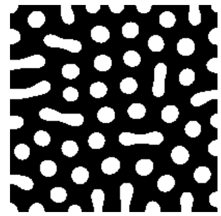


Figure 4. Spot pattern

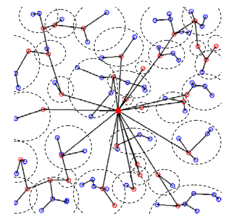


Figure 5. Clustered wireless sensor network

are different among nodes. At different rate depending on intrinsic frequency, phases shift toward one. As one of the nodes broadcasts a message, other nodes are stimulated and some of them are brought to broadcast a message. As a result of chain of stimulation, nodes are eventually merged into several groups, in which nodes broadcast a message simultaneously. Finally, the global synchronization emerges as a consequence of mutual interactions among nodes. Although delay is not considered in the experiment, a pulse-coupled oscillator model has been applied to a variety of network control such as clock and timer synchronization [8] and scheduling [9], [10], where communication delay exists.

2) Reaction-Diffusion Model: Next a reaction-diffusion model describes emergence of periodic patterns such as spots, stripes, and maze on the surface of animal coat through chemical interaction among cells [4]. In a reaction-diffusion model, two hypothetical morphogens, i.e., activator and inhibitor, are considered. The activator activates both of the activator and inhibitor to increase their concentrations. On the other hand, the inhibitor inhibits generation of morphogens. The dynamics of morphogen concentrations is formulated as,

$$\frac{du}{dt} = F(u,v) + D_u \nabla^2 u \tag{2}$$

$$\frac{dv}{dt} = G(u,v) + D_v \nabla^2 v \tag{3}$$

where u and v are concentrations of activator and inhibitor, respectively. The first term of right-hand side of the equations is called a reaction term and expresses chemical reactions, i.e., activation and inhibition among morphogens. The second term called a diffusion term is for interaction among neighboring cells. To generate a pattern, the condition $D_u < D_v$, i.e., the speed of diffusion of inhibitor is faster than that of activator, must be satisfied.

Let us consider the field of morphogens where the morphogen concentrations are the same and stable. Now assume that small perturbation makes the activator concentration slightly larger than the inhibitor concentration at a point in the field. The increased activator activates morphogenesis generation at the point. Although both of activator and inhibitor increases at the point, the generated inhibitor diffuses around for the faster diffusion rate than the activator. As a consequence, the concentration of inhibitor at the point becomes smaller than that of activator. At the same time, the diffused inhibitor inhibits generation of morphogens and then the concentration of activator becomes small at the diffused area. Eventually, we observe the spatial distribution of nonuniform morphogen concentrations, i.e., a pattern.

Autonomously generated patterns can be used in several network controls where a pattern of communication and control appears, such as routing, clustering, and placement of nodes, agents, or contents. For example, a spot pattern generated by the reaction-diffusion model in Figure 4 resembles to the clustered structure of a wireless sensor network in Figure 5. In [11], a node evaluates the reaction-diffusion equations by using the morphogen concentrations of itself and neighboring nodes. Eventually a spot pattern appears where each spot is centered at a node with the highest activator concentration in the proximity, which becomes a cluster head. Neighboring nodes, i.e., cluster members, send their sensor data following the gradient of activator concentration to a cluster head. By taking account of the residual energy in the morphogen concentrations, energy-efficient clusters can be formed in a self-organizing manner. It is shown that the near-optimal clusters can be formed by a reaction-diffusion based mechanism with only 8.4% increase in the energy×delay cost from the optimal solution. Another example of applications of a reaction-diffusion model is scheduling of spatial TDMA MAC protocol [12].

3) Response Threshold Model: A response threshold model explains division of labor in a colony of social insects [5]. The ratio of individuals engaged in a certain task is autonomously controlled in accordance with the demand. The demand s of a task changes as,

$$\frac{ds}{dt} = \delta - \frac{\alpha N_{act}}{N}, \tag{4}$$

where δ corresponds to the per-time increase in demand, N is the total number of individuals, and N_{act} means the number of individuals engaged in the task. When N_{act} is not sufficiently large, the demand increases. The probabilities that individual i starts or stops performing the task are given as,

$$P(x_i = 0 \to x_i = 1) = \frac{s^2}{s^2 + \theta_i^2}, P(x_i = 1 \to x_i = 0) = p. \quad (5)$$

x_i indicates the state of individual i, where $x_i = 1$ corresponds to performing the task. θ_i is a response threshold of individual i against the task, which implies the willingness or hesitation in doing the task. p is a constant. Adaptive division of labor or specialization emerges from the following learning function.

$$\frac{d\theta_i}{dt} = \begin{cases} -\xi & \text{if } x_i = 1 \\ \varphi & \text{if } x_i = 0 \end{cases} \quad (6)$$

This adaptation leads to division of labor in two groups, specialists actively participating in task having a small threshold and idle ones having a large threshold. When some of specialists accidentally die, the demand begins to increase. Then, individuals belonging to the latter group eventually start to perform the task. Finally the appropriate ratio $\frac{N_{act}}{N}$ recovers.

Examples of application of a response threshold model include task allocation for mobile sensor network coverage [13] and sensor and actuator networks [14]. In II-C6, we incorporate the response threshold model with the pulse-coupled oscillator model to achieve an energy-efficient and adaptive surveillance control.

4) Attractor Selection Model: Finally, an attractor selection model duplicates non-rule adaptation of *E. coli* cells to dynamically changing nutrient condition in the environment [6]. A mutant *E. coli* cell has a metabolic network consisting of two mutually inhibitory operons, each of which synthesizes different nutrient, i.e., glutamine and tetrahydrofolate. When a cell generates one nutrient more, it does the other nutrient less. If a cell is in a neutral condition where both nutrients exist, the concentrations of mRNAs dominating protein production are at a similar level. Therefore, a cell synthesize either of nutrients. Once one of nutrient becomes insufficient, the level of gene expression of operon for the missing nutrient eventually increases so that a cell can live in the new environment by compensating for the insufficient nutrient. However, there is no signal transduction, i.e., embedded rule-based mechanism, from the environment to the metabolic pathway to switch between two operons.

The dynamics of concentration of mRNAs is formulated in a general form as,

$$\frac{d\vec{x}}{dt} = f(\vec{x}) \cdot \alpha + \vec{\eta}, \quad (7)$$

where $\vec{x}$ corresponds to the concentrations of mRNA. $f(\vec{x})$ is a function for chemical reaction on the metabolic network. α represents the cellular activity such as growth rate and expresses the goodness of current behavior, i.e., gene expression. Finally, $\vec{\eta}$ expresses internal and external noise

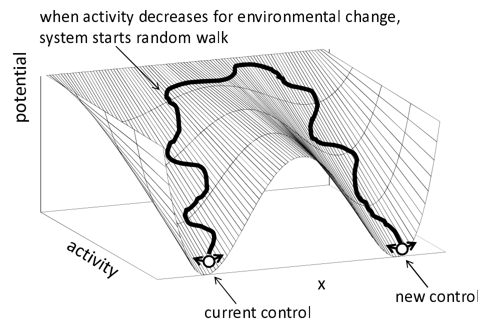


Figure 6. Attractor selection model

affecting the cell behavior. In the case of the `E. coli`'s experiments, the first term is formulated as [6],

$$\frac{dm_1}{dt} = \frac{s(\alpha)}{1 + m_2^2} - d(\alpha)m_1 + \eta_1 \quad (8)$$

$$\frac{dm_2}{dt} = \frac{s(\alpha)}{1 + m_1^2} - d(\alpha)m_2 + \eta_2. \quad (9)$$

m_1 and m_2 are mRNA concentrations and $s(\alpha)$ and $d(\alpha)$ are functions of synthesis and decomposition, respectively. Due to the mutually inhibiting relationship, the dynamics of the above nonlinear equations has two attractors, $m_1 > m_2$ and $m_1 < m_2$. An attractor is a state where a nonlinear dynamical system converges and becomes stable. If the current attractor, i.e., morphogen generation, is suitable for the current nutrient condition, the activity becomes high. Then the basin of the current attractor becomes deep and the cell stays there. Now the environment changes and the current attractor becomes inappropriate. The activity decreases and the basin becomes shallow accordingly. Being driven by the noise term, the mRNA concentrations of the cell change randomly. When the mRNA concentration corresponding to the missing nutrient becomes larger, the activity slightly increases. It makes the basin of the appropriate attractor deeper. As a consequence of entrainment, the cell eventually reaches the new appropriate attractor and adapts to the new environment (see Figure 6).

Since a new generation network would often face environmental changes and even unexpected condition, adaptation is one of fundamental mechanisms that self-organizing network controls should have. In applying to network control, $\vec{x}$ represents control parameters or control policy. When the current control is appropriate for the environment, activity α reflecting the goodness of the control becomes high and the deterministic control $f(\vec{x})$ dominates the system behavior. Once the environmental condition changes and the control becomes inappropriate, activity α decreases and relative influence of the noise term $\vec{\eta}$ becomes dominant. The system looks for new appropriate control, i.e., a good attractor, by being driven by random and stochastic control. Eventually the system finds and reaches a new good attractor.

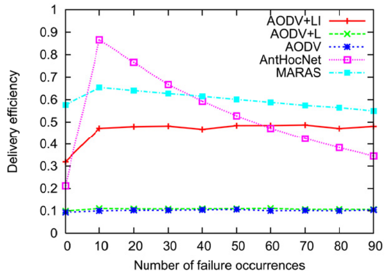


Figure 7. Performance comparison among ad-hoc routing protocols

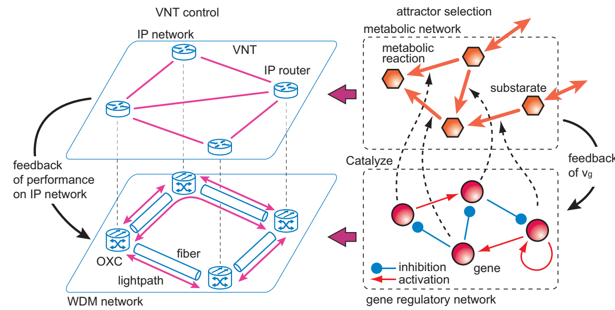


Figure 8. VNT control by hierarchical attractor selection

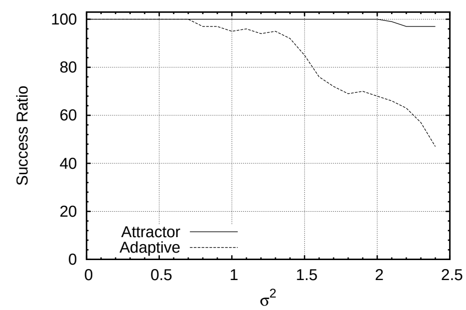


Figure 9. Adaptability of VNT control to dynamic traffic changes

An attractor selection model has been applied to multipath routing in overlay networks [15] and adaptive routing in mobile ad-hoc networks [16].

In Figure 7 a simulation result demonstrating the robustness of an attractor-selection based ad-hoc routing is shown. In our proposal, each node evaluates attractor selection equations to decide a next-hop node, to which a packet to be forwarded. Attractors correspond to neighbor nodes and the mRNA concentrations express the goodness of a neighbor node as the next hop. The activity is derived at a destination based on the number of hops that a packet travelled and it is fed back to all intermediate nodes. When a node chooses a neighbor, which contributes to establishment of a shorter path, the activity increases and a node keeps choosing the neighbor as next hop. When the path becomes longer or is broken, the activity decreases and a node begins to find a new good neighbor. In simulation experiments, 256 immobile nodes are uniformly distributed in a region of 1500×1500 m^2. Each node can communicate with neighbors within about 510 m using IEEE 802.11b. Between a fixed pair of source and destination nodes, a CBR session of 8 kbps, which sends 10 packets per second, is established. The x-axis corresponds to the number of failure occurance during a 1000-seconds simulation run. For example, with 10 failure occurance, each failure lasts for 100 seconds where randomly chosen 25% of nodes remain halted. At the end of the duration of 100 seconds, they resume operation and another set of randomly chosen 25% of nodes halt. In comparison to the standard AODV [17], AODV with local route repair feature (AODV+L), AODV with both local route repair and RREP response by intermediate node (AODV+LI), and another bio-inspired routing protocol AntHocNet [18], it is apparent that our proposal, i.e., MARAS, is more robust to failures than the others.

As one may notice, those models take the form of nonlinear temporal differential equations. It means that a system operating on self-organization engines always adapts to temporal changes in the environment. Adaptation is inherent in regular network control. In addition, no global information is required and each entity can determine its behavior by itself and in relation to neighbors. In the conventional approach, adaptation is implemented as an additional mechanism to regular network control. In the case of routing for example, next hop selection and routing table update are different and independent mechanisms. A routing table is updated on receiving intermittent control messages whereas next hop is selected on a per-packet basis.

5) Layered Attractor Selection Model: In [19], we adopt a hierachical attractor selection model of interacting a gene regulartory network and a metabolic network to virtual network topology control (see Figure 8). Genes form a gene ragulartory network of activation-inhibition relationships. A metabolic network expresses a series of production of substrates from other substrates. Chemical reaction is catalyzed by proteins, whose expression levels are controlled by genes. The dynamics of expression level of proteins is described in the form of (7), where the activity corresponds to the cell growth rate. The cell growth rate is determined as an increasing function of concentrations of substrates. A gene regulartory network adaptively and dynamically controls expression levels to achieve the high growth rate in accordance with nutrient condition.

By regarding a WDM network as a gene network, an IP network as a metabolic network, and IP-level performance, i.e., inverse of the maximum link utilization, as growth rate or activity, a WDM network adaptively and dynamically configures virtual network topology (VNT) by setting lightpaths between IP routers. Figure 9 shows a result of preliminary experiments, where the x-axis corresponds to the degree of change and the y-axis shows the probability

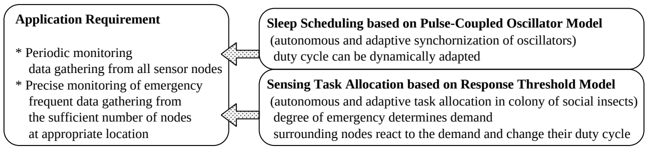


Figure 10. Self-adaptive application-oriented sensing

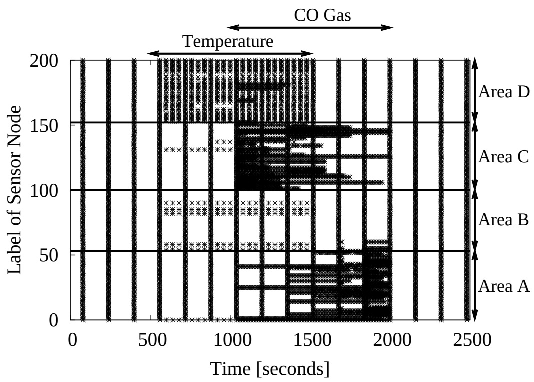


Figure 11. Adaptation of timing of message emission

that a WDM network successfully accommodates IP traffic and suppresses the maximum link utilization. As shown, our VNT control outperforms a conventional method, called ADAPTIVE, where lightpath establishment is done heuristically [20]. That is, our proposal is more robust and adaptive to dynamically changing conditions.

6) Combination of Multiple SO Engines: Now, we show an example of combination of multiple self-organization engines. Assume an application of periodic data gathering in a wireless sensor network consisting of a variety of sensor nodes, e.g., thermometer and CO gas sensor, in a plant. Under a usual condition, all sensors obtain and send their sensor data to a sink at the regular and same intervals. However, once an unusual event occurs, some sensors begin to report sensor data more frequently. The number of sensors for frequent sensing should be adapted in accordance with the degree of emergency. For example, temperature changes slowly in the order of hours and, once it becomes high, it stays high for a long period. Therefore, sensors are required to monitor temperature frequently when changes are detected, while they can decrease the sensing frequency under stable conditions. On the other hand, since gas existence itself is harmful, CO gas sensors should perform frequent sensing if CO gas exists.

In [21], taking remote surveillance of a shaft furnace in a steel plant as an application, we used a pulse-coupled oscillator model to accomplish energy-efficient sleep scheduling adaptive to sensing frequency, which is dynamically controlled by a response threshold model, see Figure 10. In Figure 11, we show how sensor nodes adapt their sensing frequency to dynamically changing sensing requirements.

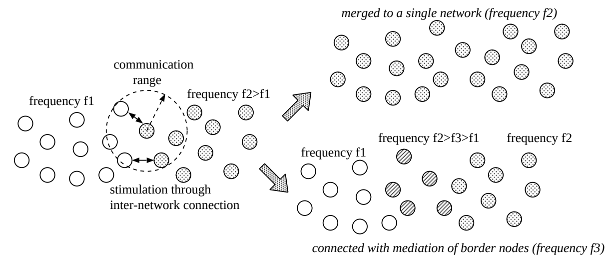


Figure 12. Fusion and connection of wireless sensor networks

200 sensor nodes are distributed in the monitored region, which further divided into four areas for the sake of explanation. Each cross indicates the time instant that a sensor node wakes up, obtaines sensor information, and emits a message. At the beginning, nothing happens in the region and sensor nodes report sensor information at the regular data gathering intervals of 160 seconds. At 500 seconds, temperature begins to increase in the area D. Detecting the increase, sensor nodes in the region D begins to operate more frequently as dense crosses show. At 1000 seconds, CO gas leakage is observed in the region C and it moves to the area A as time passes. Therefore, sensor nodes in the region first start frequent sensing. As the CO gas moves, those nodes that perform frequent sensing change as the movement of the dense area in Figure 11 indicates. At 1500 seconds, temperature stops increasing and stays high. Since there is no change in the temperature, sensor nodes, which adopt the high sensing frequency resume the normal operation.

D. Intra-layer Interaction

Nodes operating on self-organization engines directly interact with neighboring nodes by exchanging messages for stimulation in a pulse-coupled oscillator model and morphogen diffusion in a reaction-diffusion model, for example. Furthermore, they indirectly interact with each other through environmental change. The autonomous behavior of node would change the environment, by consuming the bandwidth for example. In reaction to such environmental changes, other nodes would change their behavior. Such indirect interaction induced by environmental change is called *Stigmergy* [7] and it is one of important principles of self-organization. Through direct and/or indirect mutual interaction among nodes, a network is self-organized.

Physical networks and service overlay networks also interact with each other in the physical network layer and the service overlay network layer, respectively. Direct interaction among networks is accomplished by direct message exchanges or mediation of the common network layer. In case that there is no means of direct message exchanges, e.g., communication among different node devices belonging to different networks, the common layer having multiple interfaces to those networks relays messages between them. Examples of cooperative networking can be found in some

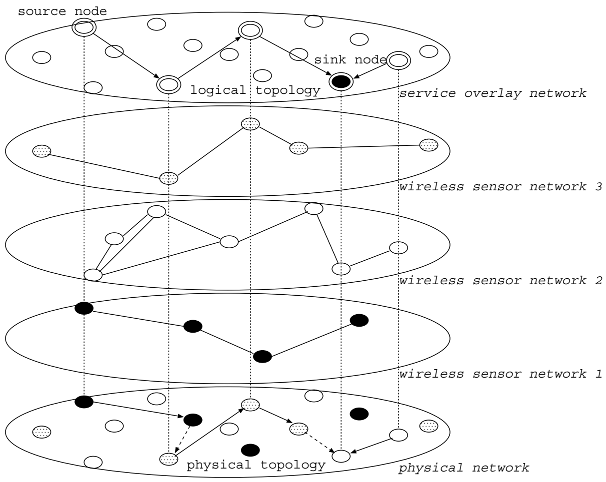


Figure 13. Layered sensor-overlay network

literatures [22], [23], [24], where networks interact with each other, they are connected with each other, and even they are merged into one depending on degree of cooperation and benefit.

For example, wireless sensor networks deployed in the same region or meeting with each other, e.g., sensor network in a room and that carried by a user entering the room, need to exchange information to provide users or applications with information or environmental control appropriate for time, place, and occasion. It is a natural assumption that they operate on the different operational frequency for energy-efficient and application-oriented control. When these networks adopt the pulse-coupled oscillator model as a self-organizing engine for frequency control, fusion, connection, and seperation of networks can easily be accomplished (see Figure 12).

E. Inter-layer Interaction

Recently especially in the field of wireless network, a concept of cross-layer design has been attracting many researchers [25]. In a cross-layer architecture, each layer optimizes its behavior taking into account information and status of other layers. For example, route establishment based on the wireless link quality expressed by the received signal strength and the amount of residual energy on nodes incorporate network layer, physical layer, and even management plane.

In the self-organizing network architecture, the common network layer allows entities belonging to different layers communicate with each other in order to exchange and share control information, get feedback from the other layer, and even control the other layer. We should note here that inter-layer interaction should be kept 'loose' not to introduce unnecessarily strong interdependency, which makes a system fragile and causes unintended consequences. For example, if an overlay network and a physical wireless network are strongly coupled with each other, the overlay network becomes too sensitive to small perturbation in the quality of wireless links and the physical topology. It changes its topology and routes actively and frequently and the resultant performance drastically deteriorates.

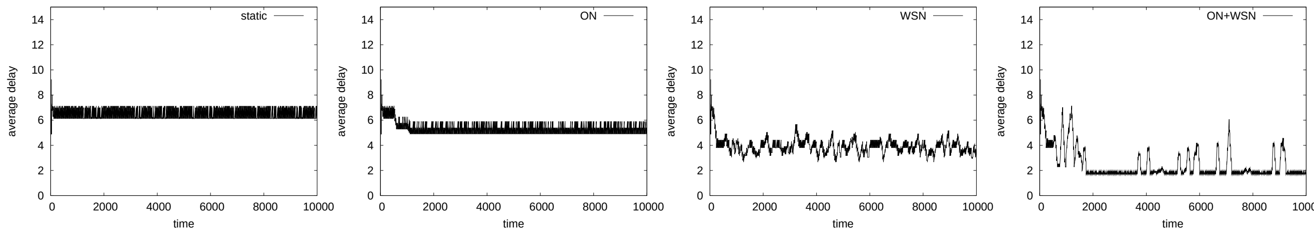
In a new generation network constructed on the layered self-organization architecture, small-scale perturbation such as local congestion, link disconnection, and node failure is handled by localized and prompt reaction of surrounding nodes. On the contrary, a network system adapts to large-scale variation, such as injection of the vast amount of traffic by flooding and spatial and simultaneous failure, by a series of reactions induced by mutual interaction among nodes and networks and spreading over the whole network, layer, and network system. Furthermore, from an inter-layer control viewpoint, influence of small-scale physical failure is absorbed in the physical network layer and hidden from the service overlay network layer. On the other hand, against large-scale physical failure, the physical network layer tries to avoid affecting the performance and control of the service overlay network layer, while the service overlay network layer adapts to changes in physical network configuration. As a result of such cooperative and self-organizing behavior, the system-level adaptability, stability, and robustness can be accomplished.

As an example of inter-layer interaction, we consider a layered sensor-overlay network. Assume that there are multiple wireless sensor networks consisting of heterogeneous sensor nodes. For the sake of energy saving, they adopt sleep control and their interval are different from each other. We consider that an overlay network is deployed over the wireless sensor networks for periodic data gathering from all or some of sensor nodes to an observatory point as illustrated in Figure 13. If all nodes involved in data gathering belong to the same wireless sensor network, the data gathering delay is the minimum. Otherwise, the delay becomes considerably large, because a node having a message to send has to wait for a next-hop node belonging to a different network to wake up. A possible way that an application can do for delay reduction without knowledge of the wireless sensor network is to adapt and find the overlay network topology leading to the minimum delay. The other adaptation in the wireless sensor network layer is synchronization. By allowing a node to addtionally synchronize with the sleep schedule of other network, the data gathering delay can be reduced very much at the sacrifice of additional energy consumption. Their adaptive behavior can be modeled by the attractor selection as,

$$\frac{d\vec{x}_O}{dt} = f(\vec{x}_O) \cdot \alpha + \vec{\eta}_O, \tag{10}$$

$$\frac{d\vec{x}_W}{dt} = f(\vec{x}_W) \cdot \alpha + \vec{\eta}_W, \tag{11}$$

where $\vec{x}_O$ and $\vec{x}_W$ corresponds to selection of overlay topology in the overlay network layer and selection of



(a) Static (b) ON (c) WSN (d) ON+WSN

Figure 14. Average data gathering delay

schedule to synchronize in the wireless sensor network layer, respectively. These layers share the same information, i.e., activity α, which is defined by the data gathering delay. Both layers behave in an adaptive manner to minimize the data gathering delay as a whole. We call this model sharing the same activity in multiple attractor selection-based controls the attractor composition model.

We evaluate the effectiveness of layered control based on the attractor composition model. 150 sensor nodes are randomly distributed in 200 m$\times$200 m region. The communication range is set at 25 m. Each sensor node has its own intrinsic operational interval randomly chosen among 5, 10, and 15 minutes. That is, there are three groups of sensor nodes. An overlay network consists of one randomly chosen sink node and four source nodes randomly chosen among remaining 149 nodes. The data gathering interval is set at 10 minutes. Sensor data are obtained every 10 minutes at source nodes and wait for transmission.

We compare four different scenarios, i.e., Static, ON, WSN, and ON+WSN, depending on whether the attractor selection or composition is performed or not. In the Static scenario, the topology of overlay network is kept the same and sensor nodes follow their intrinsic operational intervals only. In the case of ON, an overlay network alone tries to minimize the data gathering delay by changing the logical topology. On the contrary, sensor nodes adaptively synchronize with other operational intervals in the WSN scenario. Finally, the attractor composition-based dynamic adaptation is performed in the case of ON+WSN.

Figure 14 shows the transient behavior of layered adaptation control, where the adaptation intervals of overlay network and wireless sensor network are set at 500 minutes and 50 minutes, respectively. Comparing the figures in Figure 14, it is apparent that the data gathering delay gradually decreases from the left figure to the right by introduction of adaptation mechanisms. In the case of ON and ON+WSN, an overlay network stays at a certain attractor after about 1000 minutes and 1500 minutes, respectively.

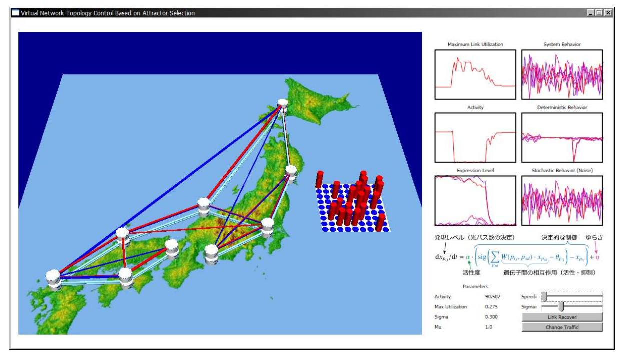


Figure 15. Screenshot of simulator

F. Evaluation Methodology

The purpose of the self-organizing network architecture is not to improve performance in terms of conventional measures such as packet delivery ratio, response time, and throughput, but to acquire higher scalability, adaptability, and robustness than ever before. However, quantitative evaluation of such *-ties and *-ness property is not trivial.

Since self-organization engines are based on nonlinear mathematical formulas, some basic characteristics such as stability, convergence, and adaptability of each control mechanism can be theoretically or analytically discussed. For example, when we consider reaction-diffusion equations of $F(u,v) = \max\{0, \min\{au - bv + C, M\}\} - du$ and $G(u,v) = \max\{0, \min\{eu + f, N\}\} - gv$ [26], the discrete step Δ in implementing the model must satisfy $0 < \Delta t < \min\{\frac{2}{d+4D_u(\Delta x^{-2}+\Delta y^{-2})}, \frac{2}{g+4D_v(\Delta x^{-2}+\Delta y^{-2})}\}$ for a pattern to converge from mathematical analysis of a bistable reaction diffusion model. In a case of the pulse coupled oscillator model, a stimulus function $\Delta(\phi_i)$ determines the speed of synchronization, but faster synchronization results in higher vulnerability to small perturbation in timer drift. However, these analytical results are obtained for a single and independent mathematical model. Basically it is not trivial to mathematically analyse and predict interaction among different self-organizing control mechanisms and their emergent

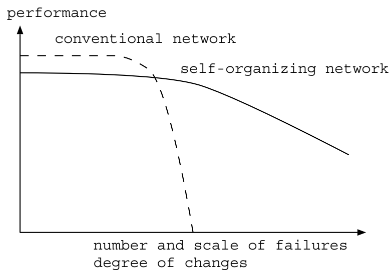


Figure 16. Comparison of self-organizing network and conventional network

behaviors. Therefore, we consider incorporating mathematical analysis for fundamental understanding of nonlinear control and simulation experiment for in-depth analysis of emergence of self-organization. For this purpose, we are now developing a novel simulator where the behavior of entities is defined by nonlinear equations and we can observe and investigate their behavior visually (see Figure 15).

Another issue is definition of range of parameters and conditions to consider. Do we need to explore the unlimited range to show the robustness against unexpected failure and condition? This still remains as future work.

III. Discussion and Future Issues

In this paper, we present the self-organizing network architecture where each of node, network, layer, and network system is self-organized through intra and inter-layer mutual interaction.

Hierarchical architecture of self-* modules can also be found in autonomic computing [27] and autonomic network [28] and there are worldwide efforts for a “clean-slate” design such as [29]. Although the main goal is the same or similar, but our architecture is different from them in organizing the whole network system by self-organization principle based on nonlinear mathematical models.

Because of self-organization, each node does not need to obtain and maintain the global information and they only need to communicate with neighbor nodes to obtain the local information. This contributes to the robustness of control [30] and the scalability where the complexity of control does not depend on the number of nodes or the size of network. Since each node only needs to calculate a set of differential equations to determine its behavior, the protocol to implement is easy, simple, and lightweight.

Although our preliminary result of a specific application scenario demonstrates the superiority of our architecture, our knowledge and experience suggest that a self-organizing system is not necessarily optimal and does not always guarantee the best performance. However, we consider it is worth sacrificing performance to some extent to achieve scalability, adaptability, and robustness.

In addition to the suboptimal performance, there are some drawbacks in self-organization based control. One is that in some class of self-organization, it takes time for a system to converge and become stable. For example, as shown in Figure 3, the synchronization does not emerge at once. Therefore, the pulse-coupled oscillator model cannot be used for synchronization in the frequently changing environment, e.g., with very high mobility. Another drawback is that it would be difficult to maintain and control the whole system. Since there is no central control unit, which collects up-to-date global information, nobody knows the current status of the system. Of course it is possible to make all entities report their status to a center, it only wastes bandwidth and energy. As stated in Section I, our approach is to leave from the conventional control relying on the global or consistent information.

As Figure 16 implies, when the operational conditions stay within the expected range, a conventional and optimally designed network achieve better and even optimal performance than a self-organizing network. However, we argue that this lower performance will eventually be compensated by the advancements in network technologies, such as the increase in channel capacities and the development of new devices. Instead, we should direct our attention to the adaptability and robustness of self-organizing systems rather than their performance. In fact, self-organizing biological systems never intend to achieve the optimal performance since they slowly evolve while adapting to a dynamically changing environment. There is always some amount of spare or even idle resources left and sometimes even inefficient control can be observed. Such unused resources are the actual reason for their adaptability and robustness, and similar strategies to tradeoff between quantity and quality are also essential properties for future network technologies.

Although it is also possible for a conventional system to improve its adaptability and robustness by introducing additional and redundant network resources and more sophisticated recovery mechanisms, this would only result in slightly shifting the critical point to the right in Figure 16, which is still far below the range of adaptation capability and tolerance that a self-organizing system possesses.

Acknowledgment

This work was supported in part by National Institute of Information and Communications Technology, Japan and “Global COE (Centers of Excellence) Program” of Ministry of Education, Culture, Sports, Science and Technology, Japan.

References

[1] S. Paul, J. Pan, and R. Jain, “Architectures for the future networks and the next generation internet: A survey,” Washington University in St. Louis, Tech. Rep. WUCSE-2009-69, October 2009.

[2] N. Wakamiya, S. Arakawa, and M. Murata, "Self-organization based network architecture for new generation networks," in *Proceedings of the First International Conference on Emerging Network Intelligence*, Sliema, Malta, October 2009.

[3] R. E. Mirollo and S. H. Strogatz, "Synchronization of pulse-coupled biological oscillators," *Society for Industrial and Applied Mathematics Journal on Applied Mathematics*, vol. 50, no. 6, pp. 1645–1662, December 1990.

[4] A. Turing, "The chemical basis of morphogenesis," *Philosophical Transactions of the Royal Society of London*, vol. B. 237, no. 641, pp. 37–72, August 1952.

[5] E. Bonabeau, A. Sobkowski, G. Theraulaz, and J.-L. Deneubourg, "Adaptive task allocation inspired by a model of division of labor in social insects," in *Proceedings of Biocomputing and Emergent Computation*, 1997, pp. 36–45.

[6] A. Kashiwagi, I. Urabe, K. Kaneko, and T. Yomo, "Adaptive response of a gene network to environmental changes by fitness-induced attractor selection," *PLoS ONE*, vol. 1, no. 1, December 2006.

[7] E. Bonabeau, M. Dorigo, and G. Theraulaz, *Swarm Intelligence: From Natural to Artificial Systems*. Oxford University Press, 1999.

[8] O. Simeone and U. Spagnolini, "Distributed time synchronization in wireless sensor networks with coupled discrete-time oscillators," *EURASIP Journal on Wireless Communications and Networking*, vol. 2007, 2007, doi:10.1155/2007/57054.

[9] Y.-W. Hong and A. Scaglione, "A scalable synchronization protocol for large scale sensor networks and its applications," *IEEE Journal on Selected Area in Communications*, vol. 23, no. 5, pp. 1085–1099, May 2005.

[10] N. Wakamiya and M. Murata, "Synchronization-based data gathering scheme for sensor networks," *IEICE Transactions on Communicatios*, vol. E88-B, no. 3, pp. 873–881, March 2005.

[11] N. Wakamiya, K. Hyodo, and M. Murata, "Reaction-diffusion based topology self-organization for periodic data gathering in wireless sensor networks," in *Proceedings of Second IEEE International Conference on Self-Adaptive and Self-Organizing Systems (SASO 2008)*, October 2008.

[12] M. Durvy and P. Thiran, "Reaction-diffusion based transmission patterns for ad hoc networks," in *Proceedings of IEEE INFOCOM 2005*, March 2005, pp. 2195–2205.

[13] K. H. Low, W. K. Leow, and J. Marcelo H. Ang, "Task allocation via self-organizing swarm coalitions in distributed mobile sensor network," in *Proceedings of AAAI-04*, July 2004, pp. 28–33.

[14] T. H. Labella and F. Dressler, "A bio-inspired architecture for division of labour in sanets," in *Proceedings of International Conference on Bio-Inspired Models of Network, Information and Computing Systems (BIONETICS 06)*, December 2006, pp. 28–33.

[15] K. Leibnitz, N. Wakamiya, and M. Murata, "Biologically-inspired self-adaptive multi-path routing in overlay networks," *Communications of the ACM*, vol. 49, no. 3, pp. 62–67, March 2006.

[16] ——, "A bio-inspired robust routing protocol for mobile ad hoc networks," in *Proceedings of 16th International Conference on Computer Communications and Networks (ICCCN 2007)*, August 2007, pp. 321–326.

[17] C. Perkins, E. Belding-Royer, and S. Das, "Ad hoc on-demand distance vector (AODV) routing," RFC 3561, July 2003.

[18] G. D. Caro, F. Ducatelle, and L. M. Gambardella, "Anthocnet: An adaptive nature-inspired algorithm for routing in mobile ad hoc networks," *European Transactions on Telecommunications (Special Issue on Self-Organization in Mobile Networking)*, vol. 16, no. 5, pp. 443–455, September/October 2005.

[19] Y. Koizumi, T. Miyamura, S. Arakawa, E. Oki, K. Shiomoto, and M. Murata, "Adaptive virtual network topology control based on attractor selection," *IEEE/OSA Journal of Lightwave Technology*, 2008, (Conditionally accepted).

[20] A. Gencata and B. Mukherjee, "Virtual-topology adaptation for WDM mesh networks under dynamic traffic," *IEEE/ACM Transactions on Networking*, vol. 11, pp. 236–247, April 2003.

[21] Y. Taniguchi, N. Wakamiya, M. Murata, and T. Fukushima, "An autonomous data gathering scheme adaptive to sensing requirements for industrial environment monitoring," in *Proceedings of 2nd IFIP International Conference on New Technologies, Mobility and Security (NTMS 2008)*, November 2008, pp. 52–56.

[22] P. Kersch, R. Szabo, and Z. L. Kis, "Self organizing ambient control space – an ambient network architecture for dynamic network interconnection," in *Proceedings of the 1st ACM workshop on Dynamic Interconnection of Networks (DIN'05)*, September 2005, pp. 17–21.

[23] E. D. Poorter, B. Latré, I. Moerman, and P. Demeester, "Symbiotic networks: Towards a new level of cooperation between wireless networks," *International Journal of Wireless Personal Communications*, vol. 45, no. 4, pp. 479–495, June 2008.

[24] N. Wakamiya and M. Murata, "Toward overlay network symbiosis," in *Proceedings of the Fifth International Conference on Peer-to-Peer Computing (P2P 2005)*, August-September 2005, pp. 154–155.

[25] V. Kawadia and P. Kumar, "A cautionary perspective on cross layer design," *IEEE Wireless Communications Magazine*, vol. 12, no. 1, pp. 3–11, February 2005.

[26] S. Kondo and R. Asai, "A viable reaction-diffusion wave on the skin of pomacanthus, a marine angelfish," *Nature*, vol. 376, pp. 765–768, August 1995.

[27] IBM Corporation, "An architectural blueprint for autonomic computing," Autonomic Computing White Paper, 2004.

[28] "Autonomic network architecture," available at http://www.ana-project.org/.

[29] AKARI project, “New generation network architecture AKARI conceptual design,” Report of National Institute of Information and Communications Technology, October 2007.

[30] M. Sugano, Y. Kiri, and M. Murata, “Differences in robustness of self-organized control and centralized control in sensor networks caused by differences in control dependense,” in *Proceedings of 3rd International Conference on Systems and Networks Communications (ICSNC 2008)*, October 2008.

Component-ware for Autonomic Supervision Services

The CASCADAS Approach

Peter H. Deussen, Edzard Höfig

Fraunhofer Institute for Open Communication Systems
Berlin, Germany

peter.deussen@fokus.fraunhofer.de
edzard.hoefig@fokus.fraunhofer.de

Matthias Baumgarten, Maurice Mulvenna

School of Computing & Mathematics
University of Ulster
Belfast, UK

m.baumgarten@ulster.ac.uk
md.mulvenna@ulster.ac.uk

Antonio Manzalini, Corrado Moiso

Telecom Italia
Torino, Italy

antonio.manzalini@telecomitalia.it
corrado.moiso@telecomitalia.it

***Abstract* — This paper describes two complementary mechanisms for the supervision of large scale and highly distributed systems structured as a cloud of autonomic computing components. The first one is based on the creation of supervision pervasions, for the supervision of clusters of components (i.e., aggregates structurally organized through one or several contracts) implementing specific services in accordance to service-specific management policies. This mechanism is designed as a supplementary service that can be requested by operational components and is structured as an ensemble of self-contained objects that implement an autonomic control loop, which does not require any a priori knowledge on the structure of the supervised system. The second mechanism promotes supervision logic embedded in the autonomic components, which exploit autonomic features and cooperate through dedicated protocols over self-organized overlay networks; this mechanism is suitable for supervising infrastructural (service-independent) functions of autonomic components, and their aggregates. The main contribution of the paper is to define those two mechanisms and to shown that they are complementary, and can be combined to achieve cross-layer supervision.**

Keywords — autonomic computing, pervasive supervision, embedded supervision, distributed systems, self-reconfiguration self-adaptation, self-organization.

I. INTRODUCTION

Networks today are composed of a wide variety of network elements that introduce a high degree of heterogeneity. The Telecommunications Management Network (TMN) is a model defined by ITU-T for supervising open systems in a communication network, implementing the fault, configuration, accounting, performance, and security (FCAPS) management areas. The TMN model can hardly meet the requirements of future trends of Telecommunication, Information and Communication Technologies (ICT) and the Future Internet (e.g., emerging of Cloud Computing as well as global pervasive environments). As a matter of fact pervasive diffusion of powerful smart devices for efficient human-computer interaction as well as increased systems heterogeneity are complicating the management and control of the whole network and service infrastructures. As such, there is a need for identifying technology and solutions to simplify the configuration and management of distributed systems whilst, at the same time, reducing the associated operational expenses. This is the main objective of Autonomic Computing [2], as argued in 2003 by IBM's homonymous manifesto. Due to the increasing complexity of large-scale computing systems, computers and applications need to be "*capable of running themselves, adjusting to varying circumstances, and preparing their resources to handle most efficiently the workloads we put upon them*" [3]. This vision took inspiration from the biological characteristics of the human *autonomic nervous systems*, where the autonomic system constantly monitors and optimizes its own status and automatically adapts itself to changing conditions.

As depicted in Figure 1, autonomous operating managers define a control loop for autonomic computing that performs functions associated to the Monitoring, Analyzing, Planning and Executing (MAPE) of processes. Autonomic managers continuously observe the managed system and its environment and handle events on which some (re-)action measures may be executed upon. Sensors and effectors provide observation and control interfaces to the managed elements. Nevertheless, in this model all autonomic "intelligence" is contained in the network of autonomic managers and in which a knowledge base encodes the know-how and practices of human operators.

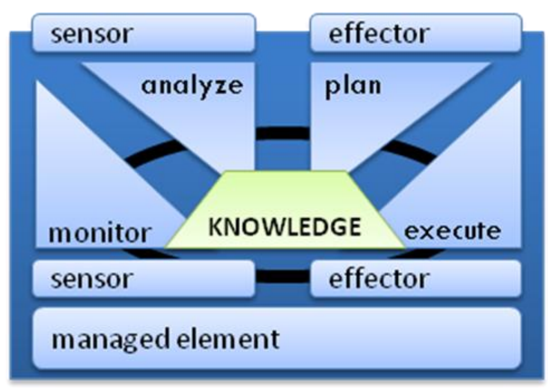


Figure 1. MAPE principle architecture (courtesy to [2]).

This paper is an extended version of the work presented originally in [1]. It outlines an approach for the supervision of highly dynamic and fully distributed systems structured as ensembles of Autonomic Components (ACs), which are based on two mechanisms: (a) a number of ACs providing a set of basic supervision services, including services aiming at predicting possible evolutions of the *system under supervision* (SUS), that dynamically self-organize to form MAPE-like control loops according to the structure of the SUS and its changes; (b) a set of supervision logic embedded in the ACs themselves exploiting autonomic features and cooperating through dedicated protocols over self-organized overlay networks. The former mechanism is mainly orientated to the supervision of clusters of ACs implementing specific services in accordance to service-specific management policies; on the other hand, the latter one is oriented to define the logic to supervise infrastructural (service-independent) functions in distributed systems that consist of large numbers of ACs, where the same service is provided by multiple instances.

The remainder of this paper is structured as follows. In Section II, the foundations of autonomic communication systems are recapitulated also introducing the notion of Autonomic Communication Elements (ACEs), developed in the context of CASCADAS Project [4], that reflect the framework the proposed supervision component-ware is implemented upon. ACEs are supposed to form services, which are configured in a self-organized way. Section III recalls two self-organization approaches, namely gossiping and rewiring, which will be important in the later course of the argument. Section IV gives a first overview over the two supervision approaches considered in this paper, namely supervision pervasions (Section IV.A) and ACE embedded supervision (Section IV.B). Supervision pervasion is addressed in detail in Section V. Its architecture, components, and interaction mechanisms are described in Section V.A. Long-term supervision is addressed in Section V.B. Section 0 describes how supervision pervasions configure themselves according to the architecture of the targeted system under supervision. An experimental framework for pervasive supervision is explained in Section VI. Section VII concentrates on ACE embedded supervision. Two applications are addressed: load balancing (Section VII.A) and power saving (Section VII.B). Section VII.C presents evaluation results for these applications. Section VIII describes how to combine pervasive and embedded supervision. Section VIII.A describes scenarios in which such a combination will be useful. Sections VIII.B and VIII.C target on a more technical level on how to use self-organization mechanisms to place supervisors, and how achieve a communication between supervisors. Section IX addresses advances beyond the state of the art. Application scenarios are described in Section X. Section XI draws conclusions and indicates further work.

II. AUTONOMIC COMMUNICATION SYSTEMS

Autonomic communication systems are composed of distributed interacting ACs, where an AC is defined to be an entity capable of sensing and adapting to environment changes whilst also performing autonomic capabilities that are related to self-CHOP (Configuration, Healing, Optimization, Protection) through the interaction with other ACs.

Although the general principles of the proposed approach on the supervision of distributed autonomic systems is independent of specific AC models, in the CASCADAS Project [4] they have been designed and experimentally evaluated, by integrating it in the CASCADS ACE Toolkit [5], by considering systems composed of several interacting ACEs. Figure 2 shows the structure of an ACE highlighting individual ACE *organs* (grey components). On the level of a particular ACE, autonomic behavior is achieved through the Facilitator, which utilizes a self-model that describes the business logic in terms of *Extended Finite State Machine*s (EFSMs) [6] capable of dealing with internal and external events, storing and accessing data and invoking task specific functionalities. Several of those state machines can be executed concurrently utilizing an asynchronous event based communication mechanism. The facilitator selects and adapts a set of those state machines in accordance to pre-defined criteria (in particular at ACE startup time) or based on incoming events from other ACEs, internal decisions made during the execution of a previous set of plans, or user interference. The executor organ is then responsible for the parallel execution of plans and their selection.

Self-models and plans implement coordination of a set of elementary ACE internal activities (i.e., they provide a “choreography” for these activities). ACEs therefore comprise a functional repository where the implementations of these activities are stored. Activities (JAVA method calls) make use of so-called *session objects* providing dynamic associative memory to store and to access data as key/value pairs. In addition to this, there is a *global session object,* which can be accessed by every currently executed plan, as well as *local session objects*, which are specific to a particular plan.

The CASCADAS approach takes the perspective that services are provided by (potentially large) ensembles of relatively simple entities realized as ACEs. The functional composition of these entities is done in a self-organized way, using the so-called Goal Needed/Goal Achievable (GN/GA) protocol as the basic means of service discovery within the ACE universe. Through GN/GA dynamic service composition is facilitated by matching a GN to available GAs, which both semantically describe the type of services or functions ACE’s desire and offer, respectively. After discovery, ACEs may establish specific contracts among each other to provide for efficient and, more importantly, secure message exchange over multilateral communication channels.

The *gateway organ* is responsible to drive the GN/GA protocol and comprises two core message types:

- Goal Needed (GN): the GN message is broadcasted to all ACEs within a certain ensemble of ACEs. Those ensembles are composed in the following way: At startup, an ACE registers itself with a broker. Several brokers form a network for achieving basic service discovery. Hence, GN messages are distributed by means of this backbone network. A GN message contains a description of a function that is desired by the sending ACE.

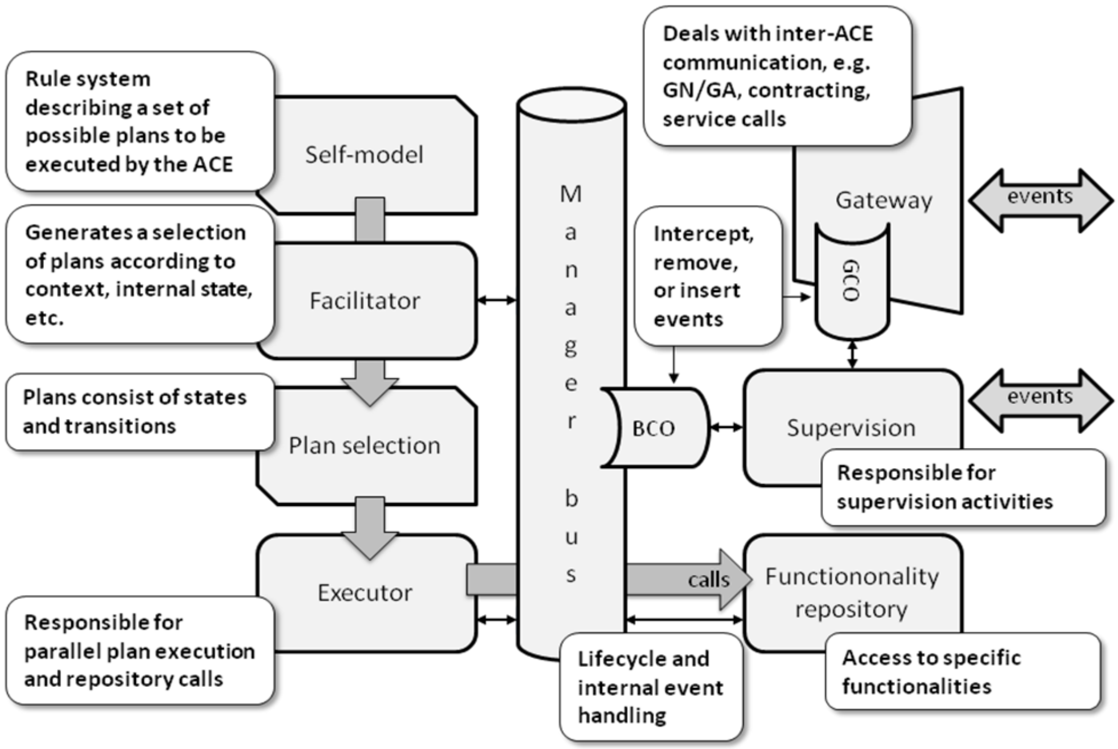


Figure 2: Autonomic Communication Element (ACE).

- Goal Achievable (GA): if an ACE receives a GN message then it compares the included functional specification with its own capabilities, and if it is capable to perform the required function, it replies with a GA (goal achievable) message containing its own address. Since the incoming GN message contains the address of its sender, the broker network is not used to transport the GA answer (the CASCADAS ACE Toolkit uses an underlying agent framework called DIET [7], which provides address-based message communication).

After the exchange of GN/GA message pairs, the initiating ACE selects a partner from the ACEs that have answered its request, and establishes a *contract*. Contracts provide a reliable multicast communication channel incorporating two or more ACEs. Within these contracts specific roles (i.e., symbolic names) are assigned to each of the ACEs involved thus providing a semantically aware way of communications. A message sent over a contract can be addressed either by a specific role (in which case the ACE assuming this role received the message), or is sent without a specific receiver role (in which case all ACEs involved in the contract would receive it). Contract establishment and cancellation is performed through the exchange of so-called contract establishment and contract cancellation events.

Hence the gateway provides the basic mechanisms for service discovery, service composition, and service internal communication. On the basis of these mechanisms, various self-organization algorithms (see Section III) have been implemented.

Before concluding the discussion of the basic ACE architecture, another ACE organ needs to be considered, namely the internal *management bus*. This bus provides ACE internal communication and coordination. Similar to inter-ACE communication, internal processes are also coordinated asynchronously by events. For instance, if the facilitator decides to establish the execution of another plan, it sends a corresponding event containing this plan to the executor. Hence, monitoring the events that travel over the management bus, provides a complete picture of the internal activities of an ACE. We will make use of this property when we describe the supervision organ later on in Section V).

III. Self-organization

The “cognitive” approaches in Autonomic Computing and Communication (for which MAPE is a paradigmatic example) are opposed by “grass root” approaches, i.e., ideas towards unmanaged self-organization of autonomic systems and services. In the CASCADAS Project, a number of self-organization mechanisms have been defined and experimentally validated to provide efficient and purpose based self-organization. A system comprising a (probably large) number of actors (e.g., ACEs) exhibits complex capabilities that emerge from the interaction of these actors. The actors itself are envisioned to be relatively simple, although no limitation on their complexity is imposed. Depending on their purpose and complexity, they possess a certain number of behavioral rules. In opposite to the cognitive approach, which has a certain flavor of centralized management, self-organization is – by definition – completely decentralized. Scalability becomes a build-in property of those systems. In this paper, we consider two self-organization approaches, namely *gossiping* and *rewiring*, which are discussed next.

A. *Gossiping*

This approach has been discussed in detail in [8][9][10]. It provides a peer-to-peer communication protocol, performed by a number of entities organized in a network. The protocol is generic in the sense that it is based on the abstract notion of a "state" and a "state update".

Figure 3 illustrates how this gossiping protocol works for active as well as passive nodes. An entity *A* may assume an *active* or a *passive* role (both behavioral alternatives are executed in parallel). In its active role, the entity waits for a *trigger* (e.g., a timeout, an external event, etc.). If the trigger is received, it selects one of its neighbors *B*, and sends its internal state *S* to *B*. In exchange, it receives *B*'s state *S*'. Then it updates its internal state using *S* and *S*', and awaits the next trigger. In its passive role, it receives information from an active entity sending its own state back also updating its internal state.

```
role active A is
  forever {
      await trigger;
      select neighbour B;
      send S ⇒ B;
      receive S' ⇐ B;
      S ← update(S, S')
  }
```

```
role passive B is
  forever {
      receive S' ⇐ A;
      send S ⇒ A;
      S ← update(S, S')
  }
```

Figure 3: Gossiping protocol [9].

A number of applications of this protocol have been discussed in [9]. To give a simple example, consider a sensor network measuring relevant environmental parameters (e.g., temperature, light intensity, etc.). To compute the average of these values and to diffuse this "new knowledge" to each element in the network, we can define a state as the currently measured parameter value, and a state update as the computation of the average of these values. It is easy to see that the state values of all elements of the network converge towards the average of all measured values.

For our purposes, we define a *state* as ACE internal state (i.e., a state of an EFSM including those session entries that are of interest for the supervision task in question).

B. *Rewiring*

Another approach for self-organization explored by the CASCADAS Project is graph rewiring [11]. In opposite to gossiping (which does not alter the structure of the underlying communication network) rewiring aims at changing the neighborhood-relationship in support of the formation of clusters of entities that are related by certain criteria. To this end, consider a match criterion between the elements of the communication graph. The idea is to alter the structure of the graph in a way that matching entities are directly connected.

Figure 4 visualizes this on a conceptual level. Here, a node *a*, acting as an *initiator*, requests a matching node from one of its neighbors *m*. This node acts as a match-maker and selects a matching candidate *b* (if such a node exists). In this case, the match-maker establishes a connection between *a* and *b* as indicated on the right side of the drawing. In the negative case, it will report back to a, which will try another neighbor as a match-maker or, alternatively, will wait until the global graph structure has been altered and a suitable match-maker/candidate pair becomes available. Note that, depending on the concrete problem to be solved by the rewiring, the edge between *a* and *m* may be deleted or may be maintained for further processing).

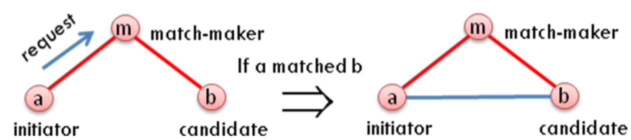


Figure 4:Rewiring

Similar to the gossiping approach, which is parametric in the notion of states and state updates, the rewiring algorithm depends on the selection of a match criterion. For instance, consider a load-balancing scenario. Here, the goal is to gradually connect all nodes that are capable to process the same type of jobs. Thus, in this case the matching criterion used is the node type. This will allow any node to, over time, find an increased number of nodes to which it can distribute the load it cannot process.

In the CASCADAS Project, rewiring has been applied to organize the clustering of ACE based services according to various criteria such as service logic, communication pattern, load distribution, fault tolerance, etc. (see Section VII). With regard to the discussion of communication mechanisms available for ACEs (Section II), the communication graph structure corresponds to contracts a particular ACE is involved in. Within the context of ACEs, graph rewiring (edge deletion and insertion) is realized through the establishment, modification, and cancellation of communication contracts.

IV. SUPERVISION FOR ACE-BASED SYSTEMS

Local and global control loops enable a component (or an aggregate of components) to react in an autonomous way to changes of the internal state and to events propagated by its environment. This feature can be fruitfully applied to implement supervision features for controlling the behavior of a component, and for actuating corrective or optimization measures when a critical situation is detected, such as a failure state, a performance problem, or a configuration error. Such autonomic capabilities should be able to address several supervision areas, such as FCAPS at different levels, from single ACEs to groups/clusters of ACEs, e.g., implementing specific services.

The approach for the supervision of distributed autonomic systems proposed in this paper is based on two mechanisms:

- *Supervision pervasions*, for the supervision of clusters (i.e., an *aggregate* of ACEs providing a common service, which is structurally organized in one or several overlapping contracts) of ACEs implementing specific services in accordance to service-specific management policies.
- *ACE embedded supervision*, for the supervision of infrastructural (service-independent) functions of ACEs and their aggregations through contracts. In opposite to supervision pervasions, which take place on the level of aggregates, embedded supervision is realized on the level of singular ACEs and their contracts.

A. Supervision Pervasions

Supervision pervasions as visualized in Figure 5 are clusters of ACEs implementing specific services according to service-specific management policies. ACE based supervision is performed through supervisors, which provide supervision as a supplementary service: As all services in the CASCADAS framework, supervisors are implemented as an aggregation of ACEs, each of them offering basic supervision functions for filtering, correlating, and elaborating events provided by the supervised ACEs, and for autonomously elaborating corrective or optimization measures. The configuration of supervisors (which is named *pervasion* because it is architecturally not separated from the SUS but *pervades* it) is dynamically set-up and updated (e.g., through self-organization techniques) to align itself to the evolution of configuration of the ACE ensemble under supervision. In this way, the ACE based supervision is able to provide autonomic control loops without any a-priory knowledge on the structure of the (ACE-based) supervised system.

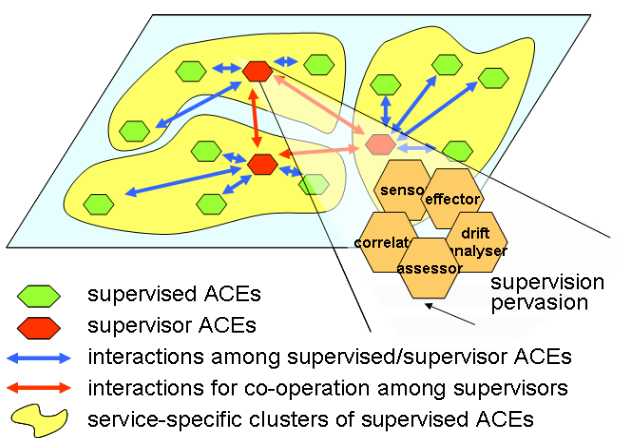


Figure 5: Supervision pervasions

The supervision service is programmable in order to implement the service-specific management policies for the monitoring and the management of groups of ACEs. Supervised ACEs are grouped into meaningful clusters, each of them controlled by a supervisor. A supervisor is able to collaborate with the supervisors of other ACEs clusters.

B. ACE Embedded Supervision

ACE embedded supervision as depicted in Figure 6 can be used to supervise the basic functions of ACEs and the interaction among them (e.g., the active contracts). This mechanism is aiming at performing supervision activities in a highly distributed way, by exploiting the self-adaptation features of ACEs, and self-aggregation of data exchanged among them. Local supervision logic, executed by each ACE, cooperates by exchanging data through a self-organized overlay [11], e.g., by means of gossiping protocols as described in Section III.A.

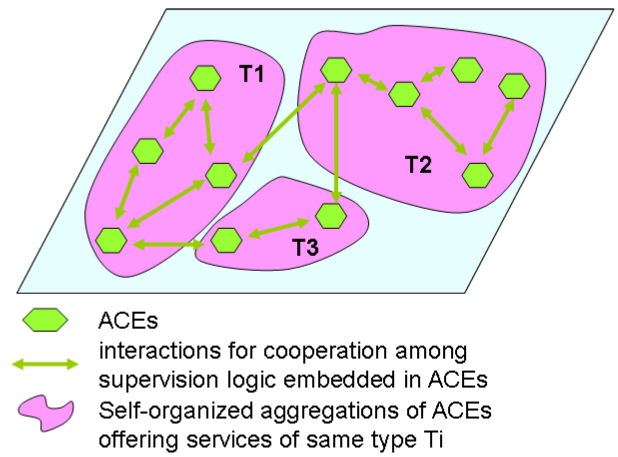


Figure 6: ACE embedded supervision

Thus this mechanism can create an approximated knowledge of the (dynamically changing) global properties of the overall system, and use them to perform local supervision decisions. This mechanism is suitable to supervise systems where multiple ACE instances provide the same "type" of services (possibly implemented differently), e.g., replicated for performance or fault tolerance reasons or deployed on end-users devices. The distribution of the logic and the interactions though overlay networks guarantee the development of scalable and robust algorithms.

As detailed in Section VIII, *ACE embedded supervision* is complementary to and synergic with *Supervision pervasion* approach. In fact, supervision pervasions are more suitable for the supervision of service specific clusters of ACEs, whereas ACE embedded supervision is oriented to the fine grain monitoring and optimization of system generic properties (such as self-repair, load distribution, and energy consumption optimization) of distributed systems. In particular, it is suitable for supervising distributed systems structured as huge amount of ACEs, where each type of service is provided by multiple instances, for instance due to redundancy and performance requirements, or as deployed on end-users terminals.

Moreover, the two mechanisms can fruitfully co-operate. For instance, a supervision pervasion must be able to react to events that are produced by ACE local supervision logic when it is not able to resolve certain situations. For examples, when the fault management supervision logic embedded in an ACE is not able to replace a failed contract for a service of a given type *T*, then it can inform its supervision pervasion, which elaborates and returns the corrective actions by e.g., reconfiguring the internal plan in order to use a service of an alternative type.

V. SUPERVISION PERVASION

Conforming to the CASCADAS architecture, supervision capabilities are realized as ACEs and as such offer a supplementary service to any ACE or ACE ensemble. A supervisor itself is an ensemble of ACEs, which are dynamically (re-)configured through service discovery and, subsequently, self-organized via the establishment of contracts, which define the relation among individual components. Thus, basic supervision functions can be provided as default services to allow for e.g., filtering and the processing of events that are produced by the supervised ACEs, and for autonomous elaboration of corrective and optimization actions.

Figure 7 depicts the architecture of a supervisor, where the use of the interaction protocols is indicated through individual arrows. For the sake of simplicity, only one instance of each component is shown, while in practice there will be always a number of supervised ACEs, sensors, effectors, etc. In addition to the basic supervision functions, supervisors may include other components in order to perform, e.g., predictions, contingency planning, etc.

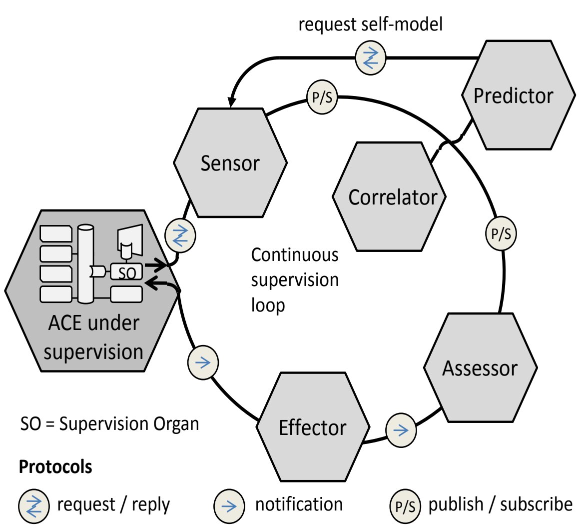


Figure 7: Organization of supervision components

This service centric perspective allows formulating and implementing supervision infrastructures, which go beyond the supervision of singular ACEs towards a more flexible and dynamic set of autonomic control loops, which are able to adjust their own structure and function to the structure of the SUS, thus forming enhanced service configurations, which are able to secure themselves against faults, performance problems, etc. To emphasize the close relationship and organizational similarity between the SUS and supervisor, those infrastructures have been named *supervision pervasions*.

Therefore, the supervised ACEs and the supervisor ACEs work synergistically realizing a supervision pervasion in the following way:

- The supervisors' autonomic behavior co-operates with and complements the autonomic behavior of the supervised ACEs.
- The structure of the supervisor is interwoven with the one of the SUS and as such is also aligned with its changes.
- Supervision can be performed based on internal and external stimuli as well as in accordance to service-specific management policies.

A. *Architecture and components*

As introduced in the previous section, the pervasive structure of a supervisor enhances the SUS by "completing" it through ACEs that implement basic supervision capabilities that are based on interfaces for observation and control between the SUS and the supervisor. As discussed in Section II, ACEs are built upon an event-driven architecture where effectively all processes are controlled by events that are propagated through the internal communication bus (for intra-ACE communication), and the gateway (for cross ACE communication, i.e., GN/GA based discovery and contract based message exchange). Hence, observing and controlling the bus and the gateway provides sufficient information to understand and to influence all ongoing processes within an ACE. Effective observation and control can be performed by interception, removal, and insertion of events sent over these communication channels.

1) *Interfacing the SUS*

The interface between the supervision layer and the SUS is realized by so-called supervision checker objects (SCOs), which are implemented as *gateway checker objects* (GCO), and *bus checker objects* (BCO) and which:

- provide basic filtering functionalities to identify events that are of interest for supervision;
- can be used to query specific information about the supervised ACEs such as its current internal state;
- provide control functionalities to steer the internal processes of ACEs;
- establish a communication channel to sensors and effectors.

SCOs can be deployed at run-time by a supervisor (the deployment process is handled on both sides by the supervision organ of an ACE; as sensors and effectors are ACEs too, they have supervision organs as well). Therefore, this mechanism provides a very flexible and generic way to set-up task specific interfaces for monitoring and actuation.

A number of functions have been implemented to provide various monitoring and actuation capabilities as summarized next:

- **Insertion**, **deletion**, and **modification** of events travelling over the management bus and the ACE gateway.
- **Interrogative requests** to retrieve the current state (the session objects), as well as the currently running plans, and the self-model of an ACE. Moreover, mechanisms are available to obtain the contracts an ACE is involved in.
- **Denial and enforcement of transition execution** within the executor. As discussed earlier, ACE plans are essentially extended finite state machines comprising states, variables vectors (i.e., session objects), and transitions that lead from one state to another while modifying variables also calling other repository functions.

2) *Communication mechanism*

As described below, a number of protocols are available for the various components to facilitate communication within an ACE based supervision pervasion:

- **Notifications** are unacknowledged messages used to distribute data between two components.
- **Request/reply** pairs are used to actively retrieve specific data.
- **Publish/subscribe** communication scheme is based on the provisioning of a certain *topic*, where a topic is a symbolic concept that is used as a category identifier for certain types of information. A supervision component, which is interested in a certain topic, broadcasts a subscription within the supervision pervasion. Any ACE providing this topic adds the requestor to a subscription list for this topic. If it obtains information matching a given topic, it distributes this information to all subscribers in the subscription list.

3) *Supervision Pervasion Components*

Sensors link the supervision system with an ACE under supervision by deploying SCOs into the supervised ACE and by establishing a dedicated communication channel for monitoring, their goal is to translate events delivered by the SCOs into the internal message format used by the supervision infrastructure, and to distribute them to other components of the supervision infrastructure, in particular to correlators and components that deal with the long term supervision of ACE's and associated services.

Correlators are responsible to aggregate monitored data from distributed sources and to correlate them with other information in order to extract meaningful indicators of the current condition of the SUS.

Predictors provide long-term supervision functions, which are discussed in more detail later on in this section.

Assessors make assumptions on the current (or future) system health based on the output of correlators, and invoke associated effectors if necessary.

Effectors are responsible to distribute contingency actions to the SCO of the various ACEs under supervision, where they are used to steer the execution of the ACE under supervision.

The application case described in Section VI uses simple arithmetic operations and pre-defined reaction patterns for analysis and actuation, but since all components mentioned above are generic and programmable, more complex correlation, assessment, and actuation approaches can be defined. For instance, the reactive part can be extended by additional components such as planners. A detailed discussion of such functionality is however outside the scope of this paper; the interested reader is referred to [12].

Note that because a supervision system is implemented by a set of ACEs, which are implemented in particular supervision organs; it is possible to extend the supervision activities to the supervision system itself by using the very same mechanisms as described above.

B. *Long-term supervision*

An important characteristic relevant for the autonomic self-evolution of pervasive services as well as their supervision mechanisms is that of prediction. If available at all levels, predictive capabilities could lead towards proactive ACE's and ACE interfaces that go some way towards the provision of calm environments, as envisaged in [13]. Such mechanisms provide the ability to predict the possible future contexts as well as interaction between stakeholders within and between ACE's. This means that, based on the observation and analysis of past behavior and the use of predictive reasoning, an ACE could predict its own future states for various aspects of e.g., its own operational environment to either guide itself to a more optimum state or, if necessary, to prevent unwanted or dangerous situations before their actually occur. Mechanisms of such supervision require a temporal aspect to be taken into account that can otherwise be discarded. That is that individual concepts and properties of a system under supervision need to be monitored over time. Similar, past behavior needs to be observed and analyzed in order to predict future situations. In relation to ACEs, relevant concepts to be analyzed include the detection of drift behavior as well as the modeling, monitoring and prediction of events, states or situations an ACE can step into or reach in the future. Thus, the general objective of long-term supervision components can be stated as to observe, model and analyze all available numerical and symbolic concepts over time, in order to predict future properties, behavior and situations of each ACE as well as any ACE ensemble. This would ultimately allow counteracting any form of behavior that could potentially lead to critical or undesired states before they are actually reached or before they occur. Additionally, it would, over time, identify the "best" execution plan for a given ACE or ACE ensemble, which can be actively used to guide new instances of a known service.

For that, three types of supervision components have been devised that are each capable of performing a long-term supervision task. Each service has been realized as an ACE itself, following the self-similar design of the ACE platform, and can be requested by a supervisor in the same fashion as any of other supervision services, i.e., via GN/GA protocol.

Drift Analyzer (DA) allow facilitating flexible long-term supervision by analyzing and forecasting numerical concepts in relation to certain boundaries a system should operate in or, alternatively, an ideal state of operation that reflects the most optimum performance of a system or a systems component. Such numerical (or ordered symbolic) properties can refer to business goals or to operational parameters of a supervised ACE and its environment.

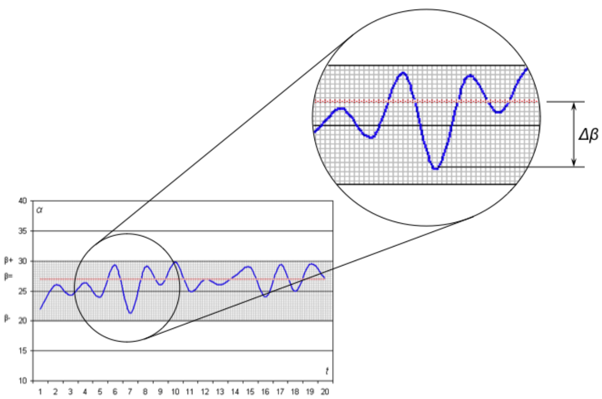


Figure 8: Drift Analyzer

The rationale of DA's is that the underlying concepts are volatile in nature and as such are likely to constantly change over time and that there is a strong desire to keep them within certain operational boundaries that reflect the correct or optimum behavior of the system under supervision. Thus, detecting if a parameter suddenly or slowly drifts towards its operational boundaries or away from its ideal state of operation would allow to deploy corresponding measures that counteract an observed effect before the system can reach a more serious, unwanted, state. As depicted in Figure 8, drift analyzers constantly monitors and analyses the numerical property it is attached too comparing it to the defined boundaries β, β^+ and the desired ideal state $\beta^=$. If drift is detected a dedicated planner component may be notified to invoke actions to counteract the detected drift.

Event Predictors (EP) predict the time window in which a certain event is most likely to occur next. As depicted in Figure 9, an EP is monitoring the occurrence of past events and computes a static as well as dynamic prediction around which a given type of event may reoccur. The static service provides the mean distance between events as its prediction, whereas the dynamic service is based on the time of calculation/request, thus taking into account the time elapsed since that last event has been registered.

This service is of particular interest for periodic services and it would allow for both, the validation of correct behavior (e.g., an event should occur periodically) or, alternatively, for the detection of fraudulent behavior (e.g., if an event occurs outside of its predicted time window). Another useful application for such a service is the priming of ACE's, ACE ensembles or the allocation of resources in anticipation of a given event. For instance, if a frequently occurring service requires specific information or a certain amount of system resources then these could be acquired or reserved, respectively, in time for the event to occur.

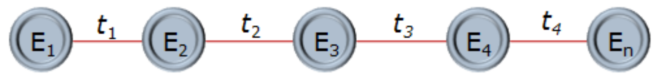


Figure 9: Event Predictor

State Predictors (SP) aim at observing and predicting the execution logic of ACEs as represented by their self-models. In particular, they allow (a) monitoring the execution of ACEs, (b) to build an execution model based on these observations and (c) based on an observed state change, to predict potential next states as well as the most likely transition(s) that lead to the predicted state(s). Note that, instead of a single possible "solution", a state predictor provides a ranked list of candidates as well as a ranked list of transitions that are associated to each candidate state. Thus, a given planner or executer component can evaluate the recommendation before executing them. Depending on their configuration, state predictors operate based on the observation of past and/or mass behavior as inspired by [14]. For instance an SP could monitor the execution of all ACEs (services) of a certain type and would, over time, construct a model that reflects how this particular type of service operates. In particular the constructed model would reveal the collective behavior of the type of service that is under supervision. If a new instance of this type of service is requested then the associated predictor component could provide recommendations of how the service should perform or behave, which would be based on the successful execution of past instances of the same service type. This would allow preventing illegal or dangerous behavior of an ACE and would also allow for the optimization of service execution in the long term. Based on the ACE concept and the associated self-model two distinct types of predictors have been devised.

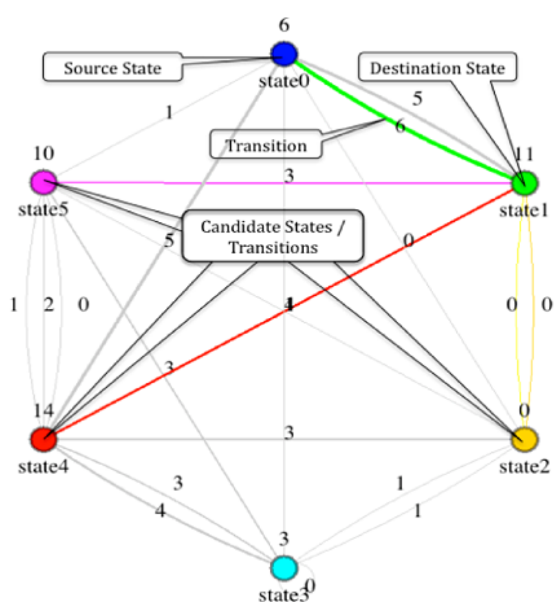


Figure 10: Meshed State Predictor (MSP)

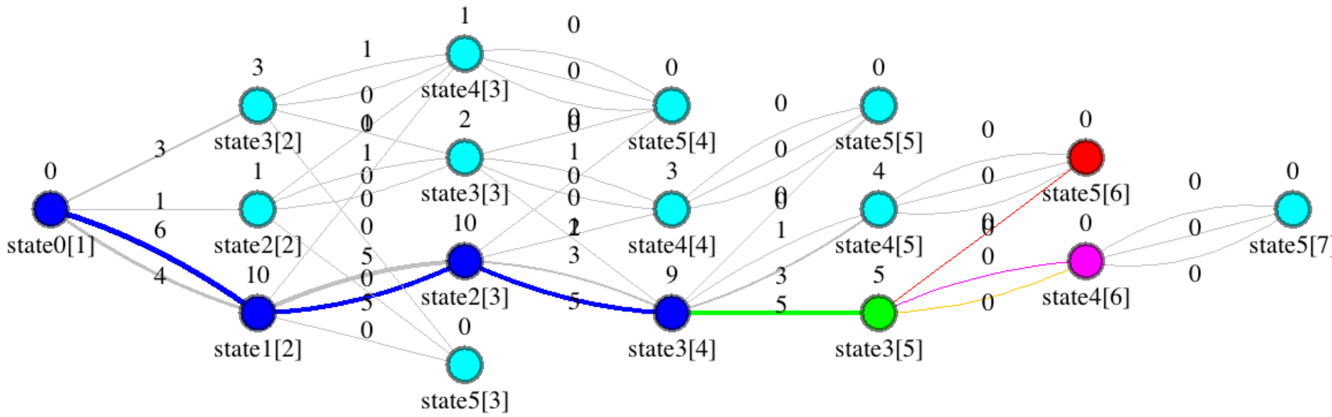


Figure 11: Directed State Predictor (DSP)

Firstly, a meshed state predictor (MSP) that only takes into account a single state change thus discarding all preceding activity and, secondly, a directed state predictor (DSP), which takes into account the entire execution path of a given service from a defined start to a defined end state. Based on this, the former is more suitable to validate stateless operational behavior as defined by individual states and transitions of a given self-model whereas the latter is more appropriate to model more specific behavior or cross ACE interactions, which are likely to be state dependent and, as such do require a more rigid model where the path of execution is relevant. For instance C can only occur if A → B has occurred first. In knowledge discovery terms this corresponds to associative pattern for the MSP and to sequential pattern for the DSP.

Figure 10 and Figure 11 show an example of the same execution model as constructed by the MSP and the DSP respectively. Each model contains a number of nodes, the transitions between them and the occurrence property that reflects how often a state has been assumed or how often a transition has been traversed. As can be seen, the full path of execution is maintained in Figure 10 whereas the model shown in Figure 10 discards this type of information. The rationale for this is based on the stateless method invocation of specific as well as common service functionality of ACE's and relates to an undirected graph in which a collection of states may form short sequences to reflect individual service execution rather than long-term business goals. In fact only the current state transition is of interest, independently of the current state of execution.

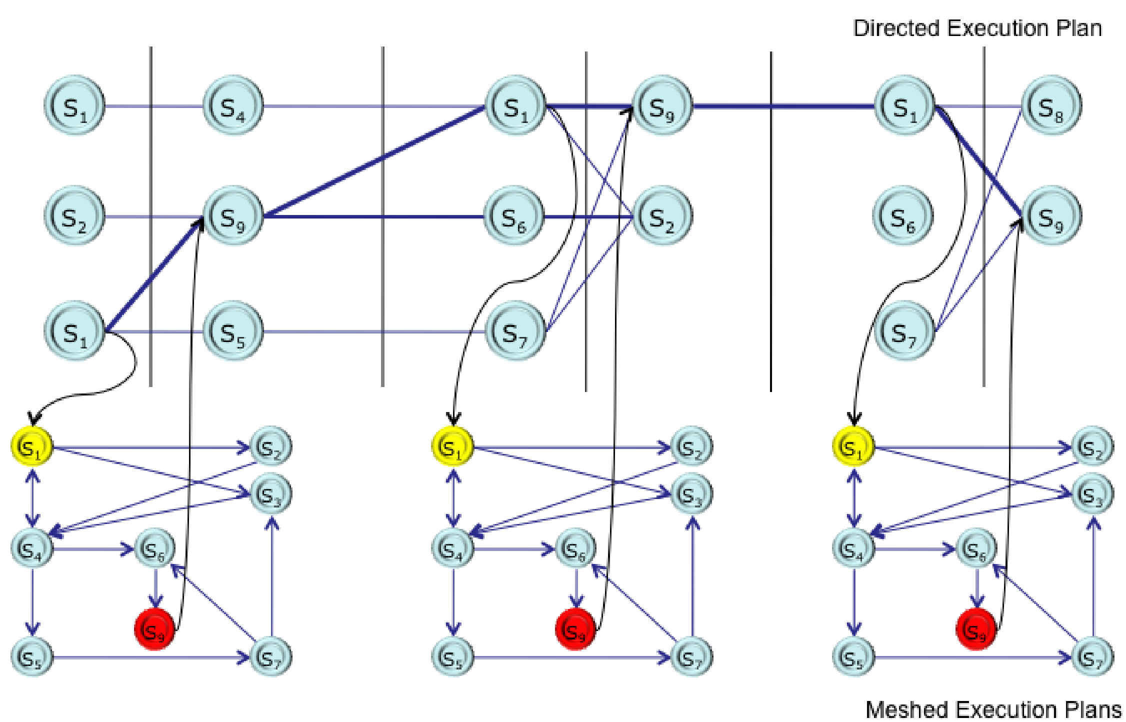


Figure 12: State Predictor Pervasion

Such a loose model of observation is ideal for short lived, stateless services where previous conditions are irrelevant. Independent of the path prefix that is maintained by the DSP, both models take into account only a single state change, which is reflected by the triple source state – destination state – transition traversed. Based on this and the properties of each state/transition, which reflect how often they have been visited or traversed in the past by the same service type, the likelihood of states / transitions to occur next can be computed. Thus an SP indicates how a service is most likely to continue based on its past behavior or based on the behavior of other instances of the same service. Such information can be used directly to, e.g., initialize subsequent states, provide system guidance, detect system violations etc.

For ACE ensembles that provide more complex services at runtime, a combination of both models is feasible that utilizes the meshed execution model at ACE level whereas the directed execution model is used for cross ACE interactions. This is depicted in Figure 12 where the top shows the execution plan of a directed state predictor, which, at certain states, steps into the execution plans of one or more meshed state predictor(s). Depending on configuration each execution plan can relate to the same or to different ACE's or ACE instances thus adapting to the distributed nature of the underling SUS. Notable for such a configuration is that no interaction between individual predictor components is required as this is embedded within the logic of the ACE ensemble that provides the overall service, which is equally reflected by the configuration of the supervision pervasion. However, how feasible such a configuration is within the context of very complex ACE configurations still needs to be explored in detail and is subject of future work.

Another aspect to be evaluated is to aid the prediction process with other parameters that relate to e.g., the current state of an ACE, its environment or its business goals. For example, during normal operation, the best path of execution of a given business process may be reflected by A→B→C with A, B and C referring to system nodes where a given service is executed on. If load on C is high, then this candidate could be demoted in favor of a candidate where load is low, e.g., A→B→D; thus ensuring that overall system load is evenly distributed. Such functionality could easily be incorporated into the state predictor by using the normalized and inverted load-factor of each node as a weight factor that influences the importance of each node within the execution model, which is normally only reflected by the occurrence property.

C. *Automatic Configuration of Pervasion*

The configuration of a supervision pervasion is done in a number of steps:

Contracting: Supervision is a supplementary service to be used by ACE ensembles that provide service(s) to a user (or another ACE ensemble). To facilitate supervision, the first step involves contracting all components (sensors, effectors, correlators, etc.) to be involved in the supervision pervasion. This is done via a special *controller* ACE, which commits a supervision contract with the SUS. Then the controller discovers, via GN/GA, the remaining ACEs, and sets up another contract for communication within the supervision pervasion. Finally, it obtains relevant configuration information, required to establish individual supervision checker components that provide a specific monitoring and control channel as discussed next. Note that the discovery of ACE's and the contracting is a service that is part of the common functionality of an ACE and as such is provided by default.

SC Deployment: Supervision checker objects (GCO and BCO) are deployed by sensors into individual ACEs that are to be supervised. To this end, a temporary contract is established between a sensor and an ACE at which an SC object is to be deployed. The SC object itself is sent as part of a specific message, which is handled by the supervision organ of the ACE to be supervised. After deployment, each SC object establishes a connection to a sensor as well as an effector ACE.

Subscription: A publish/subscribe based interaction mechanism is used as a general communication paradigm within the supervision pervasion. For instance, correlators as well as state predictors subscribe to information *published* by sensors, where the specific selection of topics obviously depends on the SUS and on the supervision task to be performed. Hence, the publish/subscribe protocol provides a data-flow driven group communication schema, where groups are defined by topics.

Re-configuration: Changes in the architectural structure of the SUS can be detected in several ways. The most generic approach is to use the BCO of an ACE to intercept events that steer the reconfiguration (contract cancellation, discovery, new contract establishments, etc.) on the internal communication bus, and to forward this information via sensors to a dedicated correlator. In some cases it is however easier to simply notify the supervisor ACEs about an ongoing reconfiguration, which in turn will adapt to this change.

The supervision pervasion reacts to the reconfiguration of the ACE ensemble under supervision by performing reconfiguration operation on itself. In particular it removes SC objects from ACEs that are not longer part of the SUS, and deploys new SCs to ACEs that are part of the new SUS ensemble. Moreover, it adapts its internal structure to reflect the new architecture of the supervised ensemble using the mechanisms (contracting, subscription) described above.

Termination: Supervision activities are terminated (or suspended in the case of long-term supervision) when the ACE ensemble under supervision decides to break the supervision contract, which is usually the case when the service contract grouping this ensemble is terminated. The controller ACE notifies all components of the supervision pervasion, and breaks the contract between them. As for the long-term supervision components, a contract can be re-instantiated to the same statefull supervision object thus allowing for the continuing observation of execution plans.

VI. EXPERIMENTAL FRAMEWORK FOR SUPERVISION PERVASION

A prototype of the proposed pervasive supervision mechanism has been implemented. It is integrated within the CASCADAS ACE Toolkit and it is available as open source [4]. Within the prototype, each of the supervision components is realized as a separate ACE. In particular, the supervision library includes a set of generic ACEs, one for each component thus providing basic as well as long-term supervision features (i.e., sensor, effector, correlator, assessor, drift analyzer, planner, state predictor, event predictor). In order to accommodate for systems specific requirements, these components can be further specialized by extending these components. More importantly, the supervision library provides relevant communication and interaction protocols (request/reply, notification, publish/subscribe) to set-up and (re-)configure a supervision pervasion and its components.

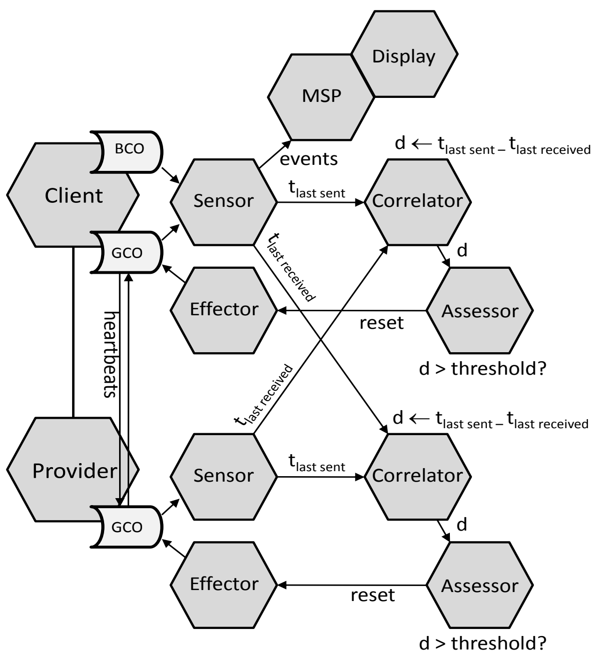


Figure 13: Dynamic Reconfiguration Scenario

The described supervision approach has been applied to supervise a video service implemented as a set of distributed ACEs. Pervasive supervision has been introduced in order to handle failures of ACEs implementing the video client and one of several available video providers as depicted in Figure 13. The goal for this scenario is to autonomously reconfigure the supervision pervasion (that is the SUS and the supervision components) if the Provider-Client relationship develops a fault.

Subject of such supervision activity is the liveliness of the contract between the client ACE and the provider ACE. Supervision is done by issuing an exchange of heartbeat signals between these two ACEs, hence, if the contract is malfunctioning in one or both directions, this fault can be detected by comparing the time stamps of sending and receiving a heartbeat signal. Heartbeats are handled by GCOs that are deployed into the supervised ACEs. Hence, the liveliness validation mechanism is transparent to the supervised ACEs where dynamic reconfiguration of the supervision pervasion takes place if the provider changes. In this case, the scenario depicted in Figure 13 is automatically adapted to work with the new service contracts. Relevant functions to inject specific GCOs / BCDs into the new SUS are provided via the ACE framework.

For long-term supervision, the MSP computes the probability of subsequent states based on observed state changes within the execution logic of the SUS (in this case, the video player ACE). Hence, in the above scenario predictions are related the probability that a certain channel will be selected and the probability that a fault during transmission will occur due to a contract problem. These probabilities are computed on the basis of sending the events of self-model transitions, which occur when the channel changes or timeouts for individual video channels occur, to the MSP ACE. The former is of particular interest, as it would allow a system to determine the channel that has been selected most in the past, which in turn could be selected if the currently selected channel becomes unavailable or if the selection procedure develops a fault.

We have not yet carried out a detailed analysis of the performance of the pervasive supervision approach. For general ACEs however, such an analysis is available, which allows inferring results also for the specific application described in this paper. Resource consumptions can be expressed in several terms:

Number of Thread: The number of threads an ACE runs in stand-by mode is around 11. Changing from an idle to a working mode the number of threads is increased to 18, with additional three threads for each new contract (which are needed to handle incoming and outgoing messages). Moreover, each parallel plan executed by an ACE adds another thread. An ACE involved in a supervision pervasion contributes to two to five contracts and runs two to six concurrent plans. Hence, in terms of threads, the performance burden seems to be significant. It should however be noted that a comparable supervision system, which is not based on ACEs would require resources too. In particular, the establishment of communication relations (corresponding to contracts) and working threads (corresponding to parallel plans) would be comparable. The main difference is that those threads are executed by ACEs and not by an external supervisor.

Memory Consumption: According to the experimental evaluations performed on the CASCADAS ACE Toolkit [15], the memory consumption turns out to be largely linear compared to the number of ACEs a system comprises and the number of established contracts (i.e., the memory needed for input queues). Hence, similar remarks on a comparison with supervision systems that are not ACE based apply.

A detailed analysis of response times in relation to the current application case is not yet available. Nevertheless, SCOs have been designed to infer with the internal event propagation of an ACE in the least possible way (monitoring is non-blocking). Hence, the performance burden added to an

ACE by deploying SCOs can be expected to be not significant.

To summarize, the impact of adding a supervision pervasion to an ACE configuration depends clearly on the architecture of this pervasion. Implementing feedback loops for each ACE (or pair of ACEs as in our application example) by means of a complete supervision configuration clearly increases the resources used in an unreasonable way. Supervision pervasions become (at least in term or resources) meaningful if larger configurations of ACEs are considered as SUS. For micro-level supervision the approach described in the following Section VII is more appropriate.

VII. ACE EMBEDDED SUPERSION

ACE embedded supervision is based on the self-adaption of behavior according to local "supervision" logic. It processes their internal state (e.g., active contracts, load information), and information received from their neighbors in "supervision" overlays. These overlays are used to communicate (through gossiping protocols) information about given ACEs that are relevant for distributed supervision algorithms. They are continuously adopted to efficiently achieve interactions among local supervision logic and as such guarantee scalability by keeping the interactions local thus avoiding information flooding.

As depicted in Figure 14 two types of "supervision" overlays are used to facilitate ACE embedded supervision:

- Achieving (T): interconnecting the ACEs, which provide a service of type T.
- Contracting (T): interconnecting the ACEs with an active contract to a service of type T.

The construction of the "contracting (T)" and "achieving (T)" clusters makes use of the rewiring algorithm described earlier in Section III.B.

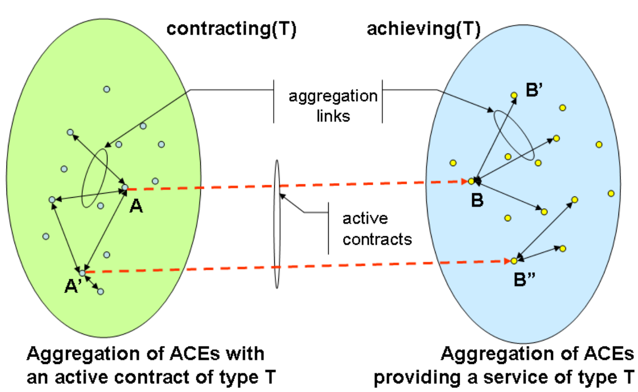


Figure 14: Self-organized overlays for service type T

This algorithm relies on the possibility to reconfigure the contracts of a given SUS and implies that services are state-less. If an execution context is needed to achieve multiple-request transactions then it can be passed as an argument within the request and response message. In this way, an ACE A with an active contract to another ACE B offering a service of type T, can replace B with another ACE that provides a service of the same type T, without losing the current context of execution. That is that information relevant for the current execution of a service are maintained, which allows such a service to continue execution instead of restarting it.

To achieve this, the ACE logic is enriched with specific supervision logic (described as a set of self-models), which process the ACE internal state and the information exchanged with the neighbors of the supervision overlays.

The following sections describe and evaluate ACE embedded supervision algorithms for load balancing, and power saving. An additional algorithm for handling contract failures outside the scope of this paper but is described in detail in [16].

A. ACE Load Balancing

The Load Balancing algorithm (LB) implements load distribution policies, in a fully distributed way. LB enables the migration of load, in terms of contracts, from ACEs that are overused to ACEs that are underused. Such underused ACEs "invite" their neighbors to the achieving (T) overlay in order to redirect some of their contracts, which is depicted in Figure 15 and facilitated as follows:

- when B' is underused, it informs all its neighbors in achieving(T);
- if B, one of B' neighbors, has a high load, it replies to B' by accepting the offer, and, negotiates with B' the load to be transferred. This negotiation mechanism effectively prevents B to transfer too much load to B';
- when A sends a request to B through one of the contracts that have been transferred to B', B informs A to redirect its contract to B'. Then A destroys the contract with B and send to B' the request to create a new contract.

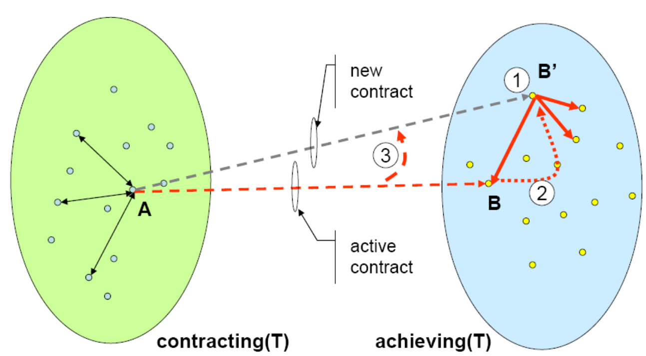


Figure 15: Load balancing supervision logic

B. Power Saving

The power saving algorithm extends the LB algorithm and assumes that each node in a distributed computing cloud is associated with an ACE that is in charge of its supervision, and as such of its optimized use within the cloud. This supervisor ACE is able to monitor the load of the node and the contracts to the services it provides. The logic stems from the fact that a node in stand-by consumes much less energy than a node that is idle. Moreover, the energy consumed by an active node is proportional to its load, but with a small difference between an idle and a fully loaded state [17]. Thus, a group of servers with low utilization is a waste of energy considering that the same work could be executed by a smaller number of servers. In this case, the remaining servers could be put in stand-by, thus reducing energy consumption.

An ACE A supervising an underused node could contact its neighboring nodes in achieving (T) to take its entire load. If this succeeds, the node monitored by A can go in stand-by to save energy. Vice versa, it could be woken-up by a node that has a high workload if this node is not able to find any currently awake node with sufficient resources available.

Measures to force a node into stand-by logic may be executed by an ACE A that is monitoring an underused node, according to the following:

- if A gets a random neighbor B in the achieving (T) overlay, and
- if B is able to take all the load of A;
- then A transfers its load (contracts) to B and goes in stand-by.

Accordingly, a wake-up "call" is executed by an ACE B that is monitoring an overloaded node, which is, according to the set out LB policies, not able to transfer its load to any other node currently active. Then B selects a neighbor in stand-by mode, if any, and transfers part of its load to it.

To avoid node oscillation, a woken-up node has to wait for some period before it can go into stand-by again. Moreover, to reduce the number of failures in looking for a neighbor to wake-up, an overloaded node has to wait for a fixed time after a failed attempt in waking up a node, before performing a new one.

C. Evaluation of the algorithms

The algorithm for load balancing (LB) and its extension with power saving policies (LB+PS) have been implemented and evaluated through simulations by using the 'breve' simulation environment [18].

The simulations were executed on a set of 6400 nodes, each of which supervised by an ACE. Each node is initialized with a random number of queued tasks (in the range of 1 to 1000) and a number of contracts (ranging from 1 to 60). During each simulation cycle, each contract generated a random number of task requests (between 0 and 10), and each node executed 200 tasks.

Two thresholds were defined as follows: a node with less than 400 pending requests is considered underused, while a node with more than 2000 pending requests is considered overloaded. In order to avoid that an underused node immediately becomes overloaded, the total amount of contracts assigned to it should not exceed the number of 40 after receiving contracts from overloaded neighbors.

The energy consumption of a node is computed in "energy units", according to the following formula, which are aligned with the considerations set out in [17]:

$$\text{energy units} = \begin{cases} 8 & \text{if the node is in stand-by} \\ 120 & \text{if the node has less than 100 queued tasks} \\ 140 & \text{if the node has less than 150 queued tasks} \\ 160 & \text{Otherwise} \end{cases}$$

Due to the initial conditions, if a load balancing policy is not adopted, the system is instable, as some nodes become immediately overloaded. Moreover, in the interval [200 : 300] cycles, the experiments simulate a traffic peak with an increment of 50% of the rate of incoming tasks.

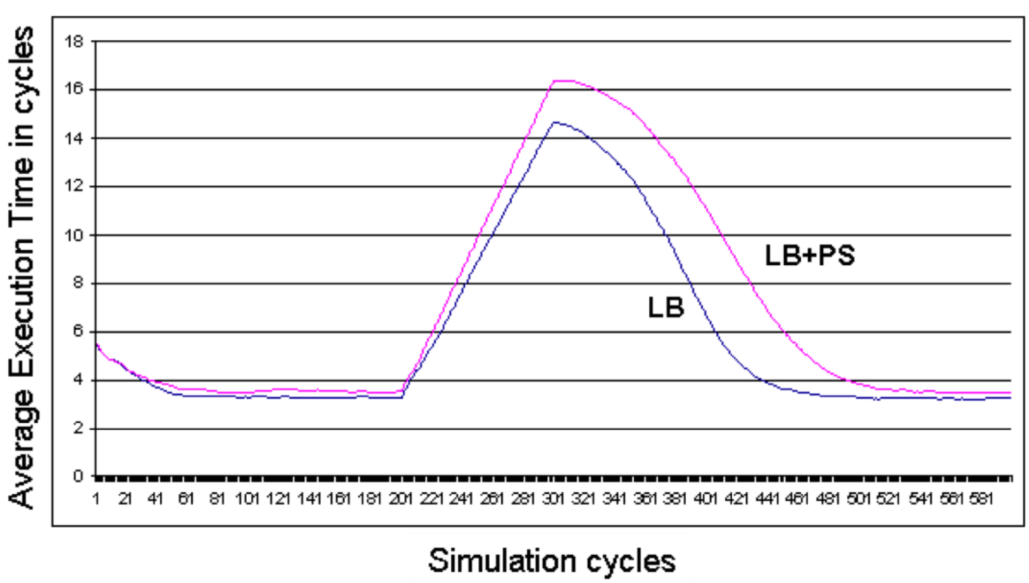


Figure 16: Comparison of task execution time

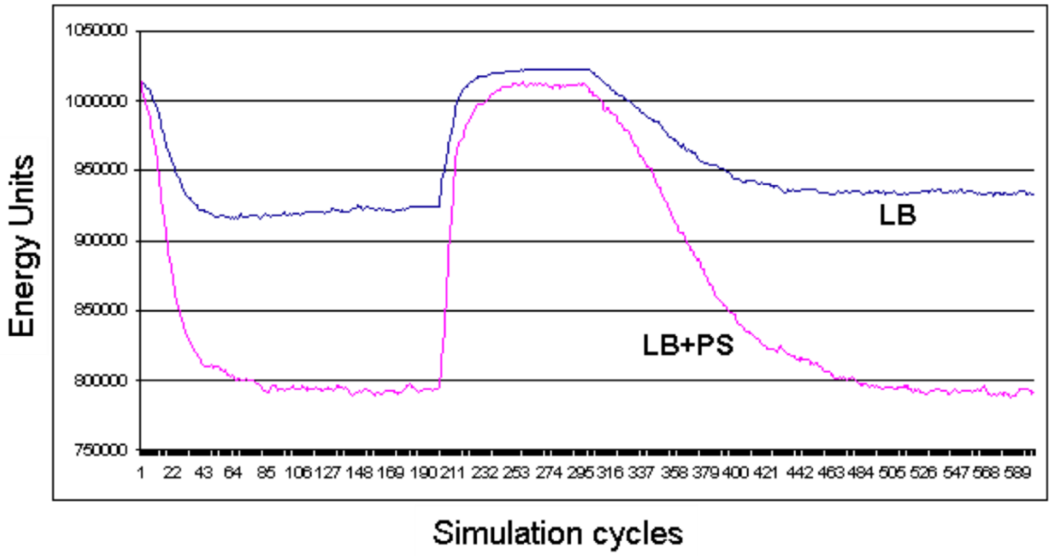


Figure 17: Comparison of energy consumption

Figure 16 and Figure 17 compare the average execution time and energy use for LB and LB+PS, respectively. As shown, the adoption of PS policies seems to introduce a benefit in the system that resulted in energy savings of about 14% (about 785000 energy units in stable state), which is achieved with a limited impact on execution time (about 5% in stable state). During the recovery phase from traffic peaks there is a maximum increment of 45% in the average execution time. This is due to a the higher delay of the system in returning to a stable state, which is based on the PS logic, which puts nodes into stand-by based on a single policy evaluation (see stand-by logic) disregarding the overall system state. Based on this some nodes may be forced into stand-by even if the overall system is still considered as overloaded. Although these nodes will be woken-up again, the system will perform inefficiently for a short time span.

LB+PS computes a quasi-optimal solution and the results were compared with the ones of a simulation of a system with an optimal distribution of load (e.g., a task is immediately assigned to an idle node). By considering the statistic variation of traffic, 5200 nodes are required to have a stable system with an average execution time of tasks of 1.95 cycles (instead of 2.15 cycles of LB+PS), and an average use of about 779000 energy units (instead of 785000). On the other hand, it is worth to point out that LB+PS requires about 25 cycles to reach a stable state in normal traffic and 160 cycles to recover from traffic peaks.

The described scenario shows that fully distributed algorithms with simple supervision logic are able to distribute load in a suitable and efficient way under normal load conditions as well as after load peaks. The efficient load balancing is also due to the fact that the proposed supervision algorithms do not move queued tasks, but contracts, i.e., the sources of requests. In this way, the algorithms balance the forthcoming load and limit the amount of information that is exchanged between ACEs. An extensive analysis of the LB and PS algorithms, and alternative options, is given in [19].

VIII. Combining ACE Embedded and Supervision Pervasion

ACE embedded supervision as well as supervision pervasions are considered to be complementary in both, the level of supervision tasks as well as their granularity. While the ACE embedded approach is more suitable for fine grained validation of system generic properties (e.g., self-repair, load distribution, and energy consumption / optimization); supervision pervasions are more suitable for service specific tasks and – due to the overhead resulting from the employment of a probably large number of supervision ACEs – applicable for tasks that can only be handled at a higher level of abstraction. For instance, the enforcement and validation of generic system management policies might turn out to be difficult using an embedded approach because it is not clear how to map those policies automatically into a local rule set or, in the context of ACEs, into the self-model. Vice versa, embedded supervision is applicable for the supervision of ACE internal properties. Nevertheless, a combination of both approaches is desirable for a number of application scenarios such as described next.

A. *Scenarios*

- Root cause analysis of faults in distributed systems is difficult because a fault might manifest itself at a completely different location of the system. While basic repair activities can be suitably handled by an embedded approach in many cases, root cause analysis requires a global view to a system, which contradicts the idea of embedded supervision to perform supervision activities on the basis of local information.
- Some problems require the coordinated effort of a number of distributed components to be solved. Consider a software update in a distributed computing network. One might want to apply a schedule that maintains a basic functionality while shutting down

some parts of the overall system to perform the update. While the execution of those coordinated activities can be performed based on local interactions and thus can be performed by a embedded supervision logic, the construction of an appropriate schedule and the distribution of sub-tasks requires again a global view. Similar, whilst updating such a system it may be important that the nodes and services from the "old" system are not mixed with the updated system. Again, this requires a more global viewpoint of the SUS.

- Supervision tasks based on statistical data concerning certain types of component requirements are derived from mass data (compare Section V.B on long-term supervision). For instance, load balancing requires the analysis of the current load situation in the neighborhood of a server. The same type of data can be used to do predictions of future load situations to e.g., identify bottlenecks. The unrestricted gossiping of such data to perform the necessary computations for such predictions is certainly not advisable; instead, the setup of dedicated event processing and correlation pipelines is required.

B. *Supervisor Placement*

In this section, we describe an approach to place supervisors "strategically" within a network of communicating nodes, and how to use it to distribute data amongst supervisors. Based on Figure 18, strategically means that:

- Each relevant cluster has to be connected to a particular supervisor. Recall that clusters are the result of the rewiring process and reflect, in a certain way, the functional structure of the SUS. Entities providing similar or connected sub-functions belong to the same clusters. In particular we are interested in those clusters that are dedicated to embedded supervision activities.
- As the system under supervision is organized according to various functional processes and considering that contracts between nodes define the interaction and relationships between interrelated processes, communication between supervisors that are related to overlapping node clusters is required for a large number of communication tasks. For instance, root cause analysis often requires the back tracing of a chain of faults until the initial problem is identified. For this, the communication between supervisors also has to back trace propagation of the fault. A similar observation applies to the software update scenario, where critical paths through the SUS need to be considered and where rollback procedures may need to be executed.

Consider an off-line computation approach where the dynamics of the underlying self-organized system, such as pre-computed distribution needs, need to be continuously updated on the basis of data obtained from the SUS. Changes in the contract structure within the SUS also require, the conceptually higher orientated, supervisor network to update itself. This shows that even off-line computation requires the existence of some kind of monitoring infrastructure that monitors and updates the current contract and clustering structure of the computing system. Self-organization mechanism could autonomously keep such a system up to date without the need for a centralized evaluation mechanism.

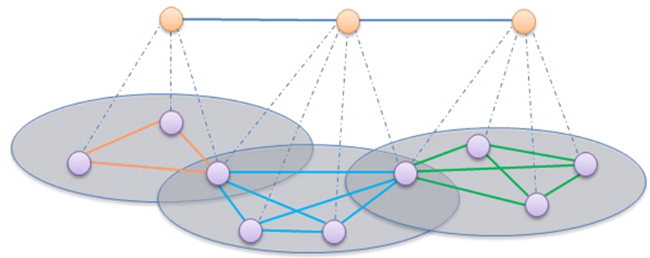


Figure 18: Strategic placement of supervisors (top nodes) in the SUS (bottom nodes)

For the initial construction of a supervisor network, we use the rewiring mechanism described in Section III.B. The network is constructed in three steps; steps 2) and 3) are continuously (concurrently) performed to update the network according to changes in the structure of the SUS. Each step is described as follows:

1. For the initial placement, a supervisor is determined for each cluster of the system. Using an ACE based system where nodes have unique identities; this can be achieved for each cluster by electing a leader (e.g., the ACE with the minimum address, or the highest computing power), and performing a GN/GA interaction to discover and to contract a supervisor ACE.
2. The supervisor discovers the other nodes within its associated cluster by using the rewiring mechanism. For that, recall that clusters are formed by the rewiring algorithm as well as by using a service dependent matching criteria. The matching criterion is that a supervisor *s* matches a node *b* if *s* is already connected to a node *a* matching *b* according to the matching criterion of the underlying cluster.
3. Finally, the connections between the supervisors that make up the supervisor network are achieved through the rewiring algorithm utilizing the following matching criterion: Supervisors s_1 and s_2 match if the sets of SUS nodes they are related to overlap.

Note that because steps 2) and 3) are performed continuously, each change in the contract structure of the SUS is detected. Thus, the supervisor network converges towards a state of strategic placement as described before.

C. *Supervisor Communication*

The gossiping algorithm described in Section III.A can be used to distribute global information about the state of the SUS. Thus, all supervisors have up to date information about the nodes in the clusters they belong too. Data related to parts of the system that are not directly connected to a supervisor are obtained through gossiping with adjacent supervisors. Naturally, the type of the data exchanged depends on the supervision task at hand. For instance, in the root cause analysis example data about "suspicious" observations can be exchanged, i.e., those information that indicate a possible propagation of the fault. In a second step, more concrete analyses can be performed through the direct cooperation of supervisors on the potential propagation path.

Hence, the supervisors are able to maintain a global picture of the state and more importantly on the organization of the underlying SUS. A mathematical approach on this idea is given in [12] and [20].

IX. ADVANCES BEYOND THE STATE OF THE ART

ACE based systems provide services by means of interactions of a probably large distributed set of ACEs with a dynamically adapted interaction structure and task diversification [21][22]. Hence, the basic assumption underlying to traditional supervision approaches (see for instance [1][23][24][25][26]), which state that the SUS maintains a static architectural structure (i.e., it does not perform run-time architecture adaptations) is not longer valid for ACE based services.

The notion of a service providing system makes a novel approach for the formulation and deployment of autonomic control loops necessary, which does not require any a priori knowledge on the structure of the SUS. In order to address this need, the pervasive supervision approach includes a novel scheme to set-up those control loops that are based on the interaction of various ACEs that form a supervision ensemble. Evidently, the structure of the supervision pervasion adapts itself dynamically to the changes of the actual structure of the SUS.

Another novel achievement is the use of a common technological basis (namely the ACE software component) both for the SUS and the supervision system, which promotes self-similarity among components. This has a number of advantages as discussed next. The introduction of additional technologies does always increase the complexity of a system; hence a reduction of operational efforts by using a supervision system that is technologically different from the SUS is at least questionable. On the other hand, for the supervision system described in this paper, a number of basic functions that are necessary for supervision are already provided by the ACE component platform itself. Examples thereof include the service discovery and contracting mechanism based on the GN/GA protocol, which supports dynamic adaptation as described earlier; The separation between the process logic (provided by ACE self-models) and the function implementation (provided by ACE functional repositories); the built-in monitoring and control mechanisms the ACE supervision organ offers. Note that generic supervision tasks (such as liveliness validation as described in the case study in Section VI) can be applied to the components of a supervision pervasion as well. Thus, self-supervision can be performed through the proposed approach.

The temporal supervision of quantitative as well as symbolic based parameters and behavior is provided as a set of long-term supervision components. A more complex supervision ensemble can be enhanced through the flexible configuration / orchestration of these components with once that offer only basic supervision features. These components have been specialized to work with the ACE model and its declarative execution logic (i.e., based on self-models). For instance, state predictors have been specifically designed to address individual features of the ACE self-model / plan philosophy to model detailed ACE behavior over time and subsequently provide detailed predictions of potential future behavior.

The proposed approach for supervision of distributed autonomic systems introduces several novelties with respect to analogous solutions based on autonomic technologies. In fact, most solutions (e.g., [23]) rely only on the self-adaptation of the components, by elaborating changes in their internal state and in their execution environment / context. The proposed approach enhances the local self-adaptation features of autonomic components, with the possibility to exchange data in a peer-to-peer fashion. Supervision-related information is exchanged with neighbors through self-organized overlays. In this way the local supervision logic can work on a local vision of the whole system. In fact, through the overlay and the self-aggregation of information, the elements are able to collect and to diffuse data from / to their neighbors, to propagate them through gossiping protocols (e.g., the ones described in [14]), and combine them with locally available data.

Self-organization algorithms have already been adopted in defining supervision capabilities. For instance, [24] describes a load balancer based on these mechanisms. Nevertheless, the CASCADAS ACE Toolkit embedded supervision goes beyond this as it is fully integrated in the abstraction and communication model. Moreover, its implementation fully exploits the ACE model, its organs, self-models and interaction mechanisms. This would allow, for instance, that the load balancing is performed at the level of contracts, and not at the level of pending tasks.

Moreover, it is important to point out that, in contrast to centralized solutions that are designed to monitor a static cluster of computing resources [25], the proposed solutions are implemented in a pervasive and distributed way across the system to be supervised and that are able to supervise systems, which are dynamically changing in the number and in the configuration of their elements.

The proposed approach for embedding supervision logic in ACEs for managing distributed systems introduces several novelties with respect to analogous solutions based on autonomic technologies. In fact, most solutions, e.g., [27], rely only on the self-adaptation of components, which is achieved by elaborating changes in their internal state and in their execution environment / context. Instead, the proposed approach enhances the local self-adaptation features of ACEs, with the local exchange of information in a self-organized overlay through gossiping protocols such as described in [8][9][10]. In order to achieve decentralized supervision logic the algorithms can create, in a fully decentralized way, an approximated knowledge of (dynamically changing) global properties of the whole system, and use them in local supervision decisions.

X. Application scenarios

In general, the supervision mechanisms proposed in this paper could be adopted to supervise any hardware and / or software system that are composed of a set of distributed and interacting components. As such, any resource of any given system can be associated to an ACE that is in charge of performing decentralized supervision logic or interacting with ACEs implementing specific supervision services.

This section elaborates briefly on some application scenarios, in the context of future telecommunications environments, where the supervision mechanisms proposed in this paper can be fruitfully exploited. For this, let's consider a simple architecture comprising the following three levels:

1. Level of resources where each of them is controlled by an ACE (or an aggregation of ACEs) through an interface, which allow its monitoring and affecting;
2. Level of ACEs supervising the resources according to embedded local supervision logic cooperating through gossiping protocols and overlays networks;
3. Level of self-organized ensembles of ACEs implementing self-adapting supervision services.

This simple architecture can be applied to tame the growing complication of supervision in future telecommunications networks that is characterized by the integration of several heterogeneous systems supporting the dynamic interconnection of huge amounts of small devices that simultaneously provide and consume services and data. The distributed supervision logic embedded in ACEs will allow the provision of supervision at infrastructure level, by coping with pervasiveness and the dynamic evolution of these environments. At the same time, the supervision pervasion would be able to self-configure and self-adapt its supervision capabilities in dependence to individual QoS and SLA specifications of specific end-to-end services.

For instance, as shown earlier, distributed supervision could improve the overall performance of pervasive clouds of computing resources by, e.g., shortening the response time through more effective load balancing policies or reducing the energy consumption by reallocating resources or putting the underused resources in stand-by. Furthermore, supervision pervasions can be adopted to cope with the management (e.g., QoS monitoring or SLA enforcement) of reliable content access and distribution services on such cloud of resources.

In fact, cloud computing is an area where the application of the proposed supervision principles can provide important benefits especially when considering that cloud computing involves a lot of disruption. For instance, the features that are essential for computing are within the network (processing and storage), the communication bus is the network itself and the input / output devices are the end user terminals. This reflects a fully distributed, highly heterogeneous system that needs to be supervised at various levels of granularity. Autonomic supervision can be used to support load balancing, dynamic configuration, fault tolerance, to enhance security, and to improve QoS in the presence of very dynamic conditions, which include resource availability and service requests. The basic idea consists in adopting ACEs to manage the high dynamicity of the cloud nodes in which users' may enter and exit the cloud in an unpredictable way increasing the dynamicity of the SUS even further. For example, each computing resource could be equipped with ACEs capable of exchanging and managing events coming from ACEs deployed on other resources and with ACEs implementing supervision services. In turn, supervision services could be used to cope with the problems of data synchronization whilst providing the proper number of duplications for the requested persistency.

Another application scenario is the supervision of distributed service provisioning platforms, where different actors can develop, provide, connect and interact, in a secure and reliable way, for selling, buying, negotiating, exchanging and trading any content, information, services and service components [28]. In such a context where components are dynamically negotiated and aggregated, supervision pervasions could be used by an actor creating a service by aggregating a cluster of components to enforce service-specific management policies, or by a provider of service components to supervise the instances of a service.

XI. Conclusions and future work

One of the most serious technological challenges of future Telecommunication, ICT and Internet endeavors will be the interconnection and management of heterogeneous systems and the huge amounts of devices that are tied together in networks of networks. Autonomic Computing has already argued that, due to the increasing complexity of large-scale computing systems, both computers and applications need to learn how to manage themselves in accordance to high-level policies as specified by human operators. Nevertheless current autonomic solutions don't exploit the real pervasive nature of distributed systems.

This paper presented a novel approach for the supervision of highly dynamic and fully distributed systems structured as ensembles of autonomic components, based on two complementary and co-operating mechanisms: supervision pervasion, and embedded supervision.

The supervision pervasion is structured as an ensemble of distributed components that implement an autonomic control loop, which does not require any a-priori knowledge on the structure of the supervised system. The architecture devised is highly modular and can be configured towards individual needs. In addition, the supervision system is able to re-configure itself according to the changes of the SUS. This mechanism is mainly oriented to the supervision of clusters of ACEs implementing specific services in accordance to service-specific management policies. It was experimentally validated by the development of a prototype, which has been made available as open source. The performance overheads can be mostly neglected considering that advantages provided. This is based on the fact that the interaction between the SUS and the supervision system is asynchronous (i.e., the supervisor does not slow down the SUS), and performance bottlenecks resulting from the introduction of a supervisor are expected to be moderate.

Full quantitative evaluation of the approach is on all aspects is, however, subject of our ongoing work. A possible evolution of the prototype would be to include the definition of the management policies through a specific language. Also, the long-term supervision components could be enhanced to facilitate the dynamic orchestration into more advanced hierarchical supervision pervasions.

The embedded supervision consists of a set of supervision logic embedded in ACEs themselves. This mechanism enhances the local self-adaptation features of ACEs, with the local exchange of information in a self-organized overlay through gossiping protocols in order to implement decentralized supervision algorithms. This mechanism can supervise potentially huge amounts of components that are pervasively distributed and interconnected by offering the capability to manage ACEs at the level of their basic functions and at the level of their aggregations. Examples are fully distributed algorithms for handling binding failures, load balancing and for optimizing the utilization of resources. Some algorithms that have been implemented according to the embedded supervision approach were evaluated by means of simulations, which showed that a quasi-optimal behavior at system level could emerge from decisions that have been made by the cooperating local supervision logic. A full integration of this mechanism in the CASCADAS ACE Toolkit [5] is planned as future work.

The two mechanisms presented can fruitfully cooperate in order to provide a “cross-layer” supervision mechanism for distributed autonomic systems. In fact, a supervision pervasion must also be able to react to events generated by local supervision logic when it is not able to properly solve a given situation. These cooperation aspects will be considered in further investigations and in future experimental evaluations.

Security has not been taken into account yet. Obviously, any automated agent that is able to monitor and to impact the execution of some system comprises a considerable security threat. A solution would be to employ standard security mechanisms to authenticate and authorize a supervisor against the supervised system. Communication between supervisor and system under supervision can be encrypted using standard cryptographic approaches. More advanced approaches could use distributed schemes to establish trust relationships as outlined, for instance, in [31][32].

ACKNOWLEDGMENT

Authors would like to acknowledge European Commission for funding the IP CASCADAS (IST-027807) (FET Proactive Initiative, IST-2004-2.3.4 Situated and Autonomic Communications).

REFERENCES

[1] Deussen P., Baumgarten M., Mulvenna M., Manzalini A., and Moiso C., “Autonomic Re-configuration of Pervasive Supervision Services - The CASCADAS Approach,” Proc. 1st International Conf. on Emerging Network Intelligence, 2009, pp. 33–38.

[2] Kephard J. O. and Chess D. M., “The Vision of Autonomic Computing,” IEEE Computer, Vol. 36, No. 1, 2003, pp. 41–50.

[3] Horn, P. "Autonomic Computing Manifesto", http://www.research.ibm.com/autonomic/manifesto/, 2001.

[4] Manzalini A., Zambonelli F., Baresi L., and Di Ferdinando A., “The CASCADAS Framework for Autonomic Communications,” in “Autonomic Communication”, A. Vasilakos, M. Parashar, S. Karnouskos, W. Pedrycz (Eds.), Springer, 2009, pp. 147–168.

[5] CASCADAS Project, “ACE Toolkit open source,” available at http://sourceforge.net/projects/acetoolkit/.

[6] Extended Finite State Machine, http://en.wikipedia.org/wiki/Extended_finite_state_machine

[7] Marrow P., Bonsma E., Wang F., and Hoile C., “DIET — A Scalable, Robust and Adaptable Multi-Agent Platform for Information Management,” BT Technology Journal, Vol. 21, No. 4, 2003, pp. 130–137.

[8] Jelasity M., Montresor A., and Babaoglu O., “Gossip-based Aggregation in Large Dynamic Networks,” ACM Transactions on Computer Systems, Vol. 23, No. 3, 2005, 219–252.

[9] Babaoglu O., Canright G., Deutsch A., Caro G. A. D., Ducatelle F., Gambardella L. M., Ganguly N., Jelasity M., Montemanni R., Montresor A., and Urnes T., “Design patterns from biology for distributed computing,“ ACM Transactions on Autonomous and Adaptive Systems, Vol. 1, No. 1, 2006, pp.:26–66.

[10] Babaoglu O. and Jelasity M., “Self-* Properties through Gossiping,” Philosophical Transactions of the Royal Society, Vol. 366, No. 1881, October 2008.

[11] Saffre F., Tateson R., Halloy J., Shackleton M., and Deneubourg J. L., “Aggregation Dynamics in Overlay Networks and Their Implications for Self-Organized Distributed Applications,” The Computer Journal, 2008.

[12] Deussen P. H., "Model Based Reactive Planning and Prediction for Autonomic Systems," Proc. Workshop on INnovative SERvice Technologies (INSERTech07), 2007, pp. 1–10.

[13] Weiser M., “The Computer for the 21st Century,” Scientific American, Vol. 265, No. 3, 1991.

[14] Han J. and Pei J., “Mining Frequent Patterns by Pattern-Growth: Methodology and Implications,” ACM SIGKDD Explorations Newsletter, Vol. 2, No. 2, 2000, pp. 14–20.

[15] Manzalini A., Minerva, R., and Moiso C., “ Autonomic Clouds of Components for self-managed Service Ecosystems,” in Journal of Telecommunications Management, Vol. 3, No. 2, 2010, to appear.

[16] Deussen P. H., Ferrari L., Manzalini A., and Moiso C., “Highly Distributed Supervision for Autonomic Networks and Services,” Proc. 5th Advanced International Conference on Telecommunications (AICT2009), 2009.

[17] Chen B., Jamieson K., Balakrishnan H., and Morris R., “Span: An Energy Efficient Coordination Algorithm for Topology Maintenance in Ad Hoc Wireless Networks,” Journal Wireless Networks, Vol. 8, No. 5, 2002, pp. 481–494.

[18] Klein J., “breve: a 3d environment for the simulation of decentralized systems and artificial life,” Proc. 8th International Conference on the Simulation and Synthesis of Living Systems (Artificial Life VIII), The MIT Press, 2002.

[19] Manzalini A., Minerva R., and Moiso C., “Supervision of decentralized systems: the magic of self-organization,” Telecom Italia Internal Report TFC0900009, 2009.

[20] Deussen P. H., "Supervision of Autonomic Systems – Tutorial," Proc. Budapest Tutorial and Workshop on Autonomic Communications and Component-ware, 2008, Published on CD.

[21] Devescovi D., Di Nitto E., Dubois D., and Mirandola R., “Self-organization algorithms for autonomic systems in the SelfLet approach,” Proc. 1st Conference on Autonomic Computing and Communication Systems, 2007, pp. 1–10.

[22] Shackleton M., Saffre F., Tateson R., Bonsma E., and Roadknight C., “Autonomic Computing for Pervasive ICT – A Whole-System Perspective,” BT Technology Journal, Vol. 22, No. 3, 2004, pp.191–199.

[23] Deussen P. H., Valetto G., Din G., Kivimaki T., Heikkinen S., and Rocha A., “Continuous On-Line Validation for Optimized Service Management,” Proc. EURESCOM Summit 2002, 2002.

[24] Garlan D., Cheng S., Huang A., Schmerl B., and Steenkiste P., “Rainbow: Architecture-based Self-adaptation with Reusable Infrastructure”, IEEE Computer, Vol. 37, No. 10, 2004, pp. 46–54.

[25] Kaiser G., Parekh J., Gross P., and Valetto G., “Retrofitting Autonomic Capabilities onto Legacy Systems,” Journal of Cluster Computing, Vol. 9, No. 2, 2006, pp. 141–159.

[26] Knight J. C., Sullivan K. J., Elder M. C., and Wang C., “Survivability Architectures: Issues and Approaches,” Proc. DARPA Information Survivability Conference and Exposition, 2000, pp. 157–171.

[27] McCann J. and Huebscher M., “Evaluation Issues in Autonomic Computing,” Proc. Grid and Cooperative Computing Workshops (GCC), 2004, pp. 597–608.

[28] Deussen P. H., Höfig E., and Manzalini A., “An Ecological Perspective on Future Service Environments,” Proc. 2nd IEEE International Conference on Self-Adaptive and Self-Organizing Systems Workshops, 2008, pp. 37–42.

[29] Di Nitto E., Dubois D. J., Mirandola R., Saffre F., and Tateson, R. “Applying Self-Aggregation to Load Balancing: Experimental Results,” Proc. Bionetics2008, 2008, pp. 1–8.

[30] Pinheiro E., Bianchini R., Carrera E., and Heath T., “Load Balancing and Unbalancing for Power and Performance in Cluster-Based Systems,” Proc. Workshop on Compilers and Operating Systems for Low Power, 2001.

[31] Zubair, I., and M. H. Islam, “Adaptive Trust Management in P2P Networks using Gossip Protocol”, Proc. 4th Int. Conf. on Emerging Technologies, Rawalpindi, Pakistan, 2008, pp. 176 – 181.

[32] Cascella, R., “Enabling Fast Bootstrap of Reputation in P2P Mobile Networks”, Proc. IEEE 23rd International Conference on Advanced Information Networking and Applications (AINA-09), Bradford, UK, 2009, pp. 371 – 378.

Investigating Agent Influence and Nested Other-Agent Behaviour

Robert Logie
Department of Informatics
Osaka Gakuin University
Osaka, Japan
rob@ogu-m.jp

Jon G. Hall and Kevin G. Waugh
Faculty of Mathematics, Computing and Technology
The Open University
Milton Keynes, England
J.G.Hall@open.ac.uk K.G.Waugh@open.ac.uk

***Abstract*—Stit theory, the notion of an agent's capacity for guaranteeing – or *s*-eeing *t*-o *it* – that a state of affairs arrives, provides a rich semantic framework for manipulating agent ability. However, the syntax of stit readily allows for nesting and such nested statements are not easily imbued with meaning. In this article, and in related work, we introduce and explore a partial logical characterisation of an alternative reading for stit based on a novel interpretation of agent *influence* that allows meaningful nested agent expressions.**

***Keywords*-Cooperative systems; Stochastic logic; Stochastic automata; Adaptive systems;**

I. Introduction, stit semantics

This article builds on the work of the authors [1] [2] [3] and from the first author's doctoral research [4] in providing theoretical underpinnings for leveraging agent interaction. Although we believe the work to be more general – there appears no characteristic dependence on reactive agents – we present the work using reactive agent for simplicity's sake.

The work extends *Stit theory* ([7]), an extension of situated first order logic in which an agent's effect on its world is considered.

Our extensions stem from the observation that, in the environment of a multi-agent system, it is not unreasonable to expect to be able to reason about one agent's effect on another; previously, however, there were difficulties in reconciling the notional independence of agents with that that they can influence each other. Indeed, simply by existing together, that agents influence each other is clear – a real-world agent that occupies a spatial location prevents another agent from occupying the same location; in stit theory, this is easily interpreted as influence. In this work, in addition, we allow one agent to have influence over the *internal state* of another: as might be the case when a contract is drawn up with clauses sanctioning any agent that does not fulfil its contractual obligations.

The extension is, we think, natural and provides a means for seeing to it that agents behave in a certain manner, and does not prevent the discovery of additional behavioural complexity for an agent group. Indeed, in the scheme we describe, *coaching agents* examine group behaviours with the intent of discovering *influence extending* joint actions. An example of extended influence through joint action is the weight that two agents can carry; another is the span of a "bridge" two can form together by joining.

We explore this notion of agent influence by developing a theory based on observed agent behaviour which we then implement in a system of coached reactive agents. We then introduce two influence operators, *leads to* and *may lead to* which we then use to outline elements of a logical characterisation of influence using the semantics of stit as a template.

We extend the use of stit semantics into the analysis of systems of simple reactive agents and will allow the analysis of agent influence in complex systems (which may have systems for obligation or sanction) based on the observation of their behaviour. Although the characterisation of our theory is incomplete, we present our investigations on some logical aspects of our influence based reading for stit.

The next section in this paper provides a short outline of stit theory and branching time, and describes the difficulties that standard stit has in addressing nested other agent behaviour. We briefly describe other areas where the notion of agent influence has been used and indicate where this differs from our work. Section III introduces our theory of influence and develops it to the point where we may describe nested agent behaviour. Section IV takes the theory into a practical setting by way of a simple system of coached reactive agents. The coaching agents use observations of agent behaviour in conjunction with the theory of influence to detect nested behaviour and to synthesise new behaviours which maximise agent influence. Section V partially explores logical characteristics of our influence theory using the stit framework as a template. Finally, Section VI summarises the paper and indicates future research areas based on agent influence.

II. Background

The first semantics for dealing with agency in terms of identifying actions with what they cause were laid out by Chellas [8]. This was taken in a number of directions with Belnap and Perloff [7], and, Horty and Belnap [9] focusing on the concept of *seeing to it*. This is usually cast against a background of possible worlds, a notion developed by Kripke, set in the branching time framework proposed by

Prior [10] and developed further by Thomason [11] [12]. Branching time represents the unsettled nature of the future by offering a number of possible paths forward and it represents the settled nature of the past by offering only a single path backwards. Each of these paths is called a *history* and points where a history divides are called *moments*. We use the notion of a branching temporal frame to describe this: a frame, $\mathcal{F}$, is a strict partial order $\langle Tree, < \rangle$ over $Tree$, a nonempty set of moments. A valuation function, v, maps propositions onto moment/history pairs thus $\mathcal{M} = \langle \mathcal{F}, v \rangle$.

At any moment, m, the future can divide into two or more histories: suppose h_1 and h_2 are distinct histories and that A holds only on h_1. Clearly m alone is insufficient for evaluating the future value of A, this most be done against a moment and history pair thus we say that A holds on m/h_1 but not on m/h_2. A proposition A is true when the valuation function indicates: $\mathcal{M}, m/h \models A$ iff $m/h \in v(A)$.

As agents makes choices, histories branch. As such, agent choices can be thought of as partitioning the set of possible futures passing through the moment where agents make that choice. Following Horty and Belnap [9] we assume a choice function which maps each agent, α on to a partition, $Choice^m_\alpha$, of all of the histories, $H_{(m)}$ through a moment m. Figure 1 shows a single moment, m, where an agent, α can choose between three actions J, K and L. Each action is distinguished by the histories it generates: L leads to a single history h_5, whereas both J and K lead to two each (presumably because of some phenomena hidden to us). The collection of histories at moment m for α, $Choice^m_\alpha$, is $\{\{h_1, h_2\}, \{h_3, h_4\}, \{h_5\}\}$.

More formally, the agent's choices divide the five possible histories into three equivalence classes according to its choice at m of J, K, or L. If α chooses K then it admits indeterminism as it does not control which of $\{h_3, h_4\}$ is followed, which are distinguishable because A holds only on h_4. We can therefore say that α choosing K cannot guarantee A. If α chooses L then it can guarantee that the future will evolve along h_5. In this case, α's choice of L ensures $\neg A$ will hold; α choosing L exerts influence over $\neg A$.

We blur the distinction between the choice open to an agent (the J, K and L) and the equivalence class of histories that a choice admits (the $\{h_1, h_2\}$, $\{h_3, h_4\}$ or $\{h_5\}$). A statement of the form $[\alpha\ stit\colon A]$ reads that an agent, α has the ability to see to it that A holds. In Figure 1 it can be seen that A holds on all of the histories emerging from J so that despite the indeterminism of $\{h_1, h_2\}$ the agent is able to guarantee A. This simple evaluation is a Chellas stit or $cstit$ that Horty and Belnap [9] state more formally as $Choice^m_\alpha(h)$:

$$\mathcal{M}, m/h \models [\alpha\ cstit\colon A] \text{ iff}$$
$$\mathcal{M}, m/h' \models A \text{ for all } h' \in Choice^m_\alpha(h) \qquad (1)$$

Given that $\models$ indicates the relation between an m/h index belonging to some model and the formulas true at that index, equation 1 says that A must hold on all histories emerging from some $Choice^m_\alpha$, *α's* choice at moment m. Here we describe the notion of *settledness* in the branching time model. Horty and Belnap [9] indicate that settledness is historical necessity and liken it to the standard modal necessity operator $\Box$ and give an evaluation rule:

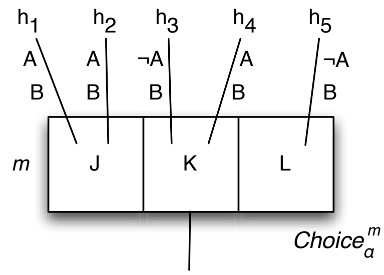


Figure 1. Example of both $[\alpha\ cstit\colon B]$ *and* $[\alpha\ dstit\colon A]$

$$\mathcal{M}, m/h \models \Box A \text{ iff } \mathcal{M}, m/h' \models A \text{ for all } h' \in H_{(m)} \quad (2)$$

This notion of necessity is incorporated into notion of *settledness* allowing us to describe propositions as being settled true or settled false at a moment and history pair m/h, following Horty and Belnap:

Definition 1: We define A as settled true at a moment, m, in a model $\mathcal{M}$ when $\mathcal{M}, m/h \models A$ for each h in $H_{(m)}$. Conversely, A as settled false at m when $\mathcal{M}, m/h \not\models A$ for each h in $H_{(m)}$.

The Chellas stit allows for but does not require that an agent be able to do otherwise and an alternative evaluation rule, the *deliberative* stit or dstit, adds this requirement, Horty and Belnap [9] state its evaluation as:

$$\mathcal{M}, m/h \models [\alpha\ dstit\colon A] \text{ iff}$$
$$(1)\ \mathcal{M}, m/h' \models A \text{ for all } h' \in Choice^m_\alpha(h) \text{ and}$$
$$(2) \text{ there is some } h'' \in H_{(m)} \text{ where } \mathcal{M}, m/h' \not\models A \quad (3)$$

Equation 3 says that in addition to Equation 1's requirement there must be at least one history, h'' in the set of all histories accessible to α at m, $H_{(m)}$, where A does not hold. The dstit reading forces the agent to make a deliberate choice Figure 1 shows that $\neg A$ holds on all of the histories emerging from L giving α a choice that guarantees $\neg A$ and providing the *negative condition* required for a deliberative stit and $[\alpha\ dstit\colon A]$ holds true at m/h_1. In contrast B holds on all histories $\{h_1, \ldots, h_5\}$ satisfying a Chellas stit but not a deliberative stit. The notion of an agent's choice at a *moment* may be extended by grouping moments horizontally across a tree, that is moments occurring at the same time, into a set known as an *instant*. This allows further refinement in the evaluation of stit expressions by applying temporal bounds. This broadens the scope of stit analysis to allow for chains of actions where one choice at a *witnessing* moment makes it possible for a later choice at the *achievement* moment in an achievement stit construct. The notion of witnessing moment is important to what follows so we state Horty and Belnap's [9] evaluation rule for an achievement stit or astit:

$$\begin{aligned}
&\mathcal{M}, m/h \models [\alpha\ astit\colon A] \text{ iff there is a moment } w \text{ such that}\\
&(1) \text{ for all } m'\ Choice^m_\alpha(h) \text{ equivalent to } m \text{ we have}\\
&\quad \mathcal{M}, m'/h' \models A \text{ for all } h' \in H_{(m')} \text{ and}\\
&(2) \text{ there is some moment } m'' \in i_{(m)} \text{ such that}\\
&\quad w < m'' \text{ and } \mathcal{M}, m''/h'' \not\models A \text{ for some } h'' \in H_{(m'')}
\end{aligned} \tag{4}$$

Equation 4 says that if $[\alpha\ stit\colon A]$ is true at m/h as a result of a choice by α at some prior *witnessing* moment then there are two requirements. The positive requirement, equation 4(1), is that as a result of $\alpha's$ earlier choice things have evolved in such a way that A is guaranteed now. The negative requirement, equation 4(2), is that at the witnessing moment A was not yet settled true so that $\alpha's$ choice at w had some real effect in bringing about A.

Definition 2: We define a witnessing moment for $[\alpha\ stit\colon A]$ as some moment w preceding m where a choice is made that allows α to see to it that A holds by a choice at m. Additionally, at moment w there must be a choice available which leads to a future where A is settled false.

A. Difficulties with stit

Consider a nested other agent stit expression of the form $[\beta\ stit\colon [\alpha\ stit\colon A]]$, i.e., β sees to it that α sees to it that A holds. This makes intuitive sense and Chellas [13] (also in Belnap et al. [14, page 275]) notes that it would be "...bizarre to deny that an agent should be able to see to it that another agent sees to something." Nested other agent expressions are syntactically well formed but their semantics fail when considered logically. When we say that $[\alpha\ stit\colon [\beta\ stit\colon A]]$ are we saying that α sees to it that β sees to it that A holds or are we saying that $\alpha' s$ action makes it the case that β is *able* to see to it that A holds? If the former reading were true then α must exercise some influence over β. Belnap et al. [14, page 274], object to this reading on the grounds that it is inconsistent and justify this stance by demonstrating a contradiction. Rather than reproduce the proof here we represent a joint choice by two agents, α and β at a moment m_0 in Figure 2. I represents β seeing to it that A holds. If β acts in such a was as to see to it that A holds then this must be the case for *all* of $\alpha's$ choices because if this were not the case then α and β would not be independent agents. Figure 2 is described by Belnap et al. [14, pages 274–275] as representing a witnessing moment, m_0, the columns represent possible choices for α and the rows represent possible choices for β. If $[\alpha\ stit\colon [\beta\ stit\colon A]]$, then, where $I = [\beta\ stit\colon A]$, I must fill a choice column for α in m_0. But because I represents a stit by β, whenever I appears anywhere in a choice row for β in m_0 it must fill that row. So I must fill the entire diagram of m_0. If so then I is by definition settled true at m_0 and this contradicts the negative requirement, described in equation 4(2), that stit statements are never settled true at their witnessing moment.

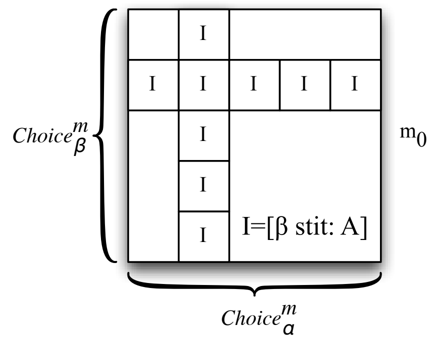


Figure 2. The logical impossibility of $[\alpha\ stit\colon [\beta\ stit\colon A]]$ (Adapted from [14, page 274])

Intuitively such statements make sense socially, α may have an obligation to β and in that way β may be thought of as seeing to it that α brings about A. Similarly β may be in a position to impose a sanction on α if α fails to bring about A. These allow for a meaningful interpretation of stit but they also bring burdens, simple reactive agents may have no notion of obligation and a system of sanctions adds an abstract choice – that of being sanctioned – for failing to bring about A. Belnap et al. [14, page 271] suggest a number of readings, including those above, that allow for a meaningful interpretation of nested other agent stit expressions but, as noted above, these may place additional requirements on agents making these approaches unsuitable for simple systems of reactive agents. When considered in the context of a society of agents, nested other agent constructs seem to imply that one agent may somehow control the other agent's choices thereby compromising its independence and agency.

The notion of influence introduced here is that one agent may influence another agent which, in turn, may use its influence to bring about A.

There have been other approaches to dealing with agent "influence". Ferber and Müller's multi agent based simulations (MABS) [15] formalism differs from this approach in that it aggregates agent actions to generate an effect on the environment. This approach, Michel [16] notes, does not readily model simultaneous actions and Weyns [17] indicates that it is limited to synchronous systems. Simonin et al. [18] extend Ferber and Müller's work into a formal design based on the B-Method. This work differs from these by treating both agents and the environment as "black boxes" and by operating on observation. Observations are used to generate hypotheses about agent abilities and these hypotheses are used with a theory of influence to generate behaviour patterns for reactive agents. Influence, in this setting, is treated as a logical property in an agent-centric stit like framework rather than something that is aggregated globally. An influence based reading of other agent expressions allows a meaningful interpretation without additional burdens, for example the cognitive ability to reason about sanctions and the extra system requirements for administering sanctions. Our influence reading allows us to extend the semantic reach of nested other agent expressions into the domain of simple reactive agents and possibly into the domain of emergent behaviour and the social systems that it is capable of building.

We consider cases of parallel action, where collocated agents act simultaneously, and serial action where agents are not necessarily collocated and do not act simultaneously. The *parallel* case is less complex because it allows agent choices to be grouped and possible futures mapped in the manner described by Horty [19, page 29]. The serial case is more complex and forms the larger part of our investigation. Where agent actions are separated temporally their choices cannot simply be grouped because the outcome of a sequence remains contingent until the final agent choice.

III. A THEORY OF INFLUENCE

We noted, in Section I, that our main interest is in simple reactive agents and the difficulty of allowing systems using these agents to improve their performance over time. How does such a simple agent evaluate its performance and, perhaps a greater problem, how may it alter its behaviour? We address these problems in two ways, we introduce an observer to the system and we allow this observer to synthesise new behaviours for agents. How far into the cognitive domain do we allow the observer to go? A cognitive observer presents subtle problems, consider a system as a state space with agents that are able to drive change in this state space. This state space will contain a number of "good" states and these in order to be achievable these good states must be reachable by at least one "route" from other states. A route, in this context, is a sequence of agent actions causing the environment to change from one state to another. A cognitive and intelligent observer may try to shape agent behaviour inappropriately because its intelligent choices may bias it towards certain behaviour patterns. This is not necessarily bad but it does not really fit with our notion of state space and routes, our notion is perhaps better characterised as a state space exploration and intelligently guided behaviour may leave parts of the state space unexplored.

This is where we introduce our notion of agent influence and we use this at the observer level Observer agents have no ability to alter the agent environment or to directly manipulate actor agents. They can generate new behaviour patterns that are placed in the environment for actor agents to acquire and this acquisition of behaviours should, over time, lead to actor agents becoming adapted so as to maximise their influence on their environment.

When an observer "knows" that a behaviour indicates that an agent has influence on some aspect of the environment then it may generate a new behaviour pattern for that agent, one that attempts to maximise the agent's use of that influential action. It is generally accepted that there are four conditions for an agent, α *knowing* that A [20, page 7]. α knows A if and only if (i) A is true; (ii) α believes that A; (iii) α has adequate evidence for A; and (iv) the relation between (i) and (iii) is not just accidental. Of the three conditions involving agents only (ii) is subjective and because of the we may quantify this subjectivity.

A. Influencing influence

The theory of influence described here follows Milner's [21] observation that "the behaviour of a system is exactly what is observable". In addition, our theory admits that in noisy environments and in cases where agent ability is contingent and not fully understood then evidence may appear to be inconsistent. Where behaviour that involves two agents is observed then it may be said that $[\alpha \ \textit{influences}\colon [\beta \ \textit{influences}\colon A]]$ without compromising the agency of either party.

Although stit semantics present difficulties in other agent settings they provide the foundation for our theory of influence. The simple reason for this is that stit semantics are rich and expressive and provide a good template for influence. Stit expresses agent ability by characterising the partitioning of possible futures according to agent choices. If an agent has unambiguous ability to bring about A then observations of its behaviour would be similar to Figure 1 where at least one choice guarantees A. We introduce a notion of *strict* stit here and use this as a basis for differentiating our influence based reading from the standard *strict* reading, we observe that:

Observation 1: Standard stit expressions have strict requirements for the truth values of propositions, if a proposition does not hold following an agent choice then a related stit expression will be falsified. Influence has weaker requirements for the truth values of propositions, a proposition not holding following an agent choice does not necessarily mean that that choice has no influence over the proposition.

The strict approach carries the implication that one agent has control rather than influence over the other agent or agents involved in a complex behaviour. It is, thus, the strict reading that presents difficulties and we intend our influence reading to be as semantically close as possible to strict stit and also to maintain the agency of all agents involved in a complex behaviour. Suppose that α*'s* ability to bring about A is contingent on another agent's choice, what would be observed then?

Consider choice K in Figure 1, it can be seen that if α chooses K, written α/K, then this leads to two possible histories, $\{h_3, h_4\}$ with A holding on one but not the other. α, it seems, can not see to it that A holds by choosing K.

Inspecting Figure 1 shows that there are histories with A and histories with $\neg A$, this means that A is neither identically *true* nor identically *false* and may, potentially, be under the influence of an agent or agents. This allows for a hypothesis that α/K has influence over A and historical data may be inspected for supporting evidence. There are three forms of evidence that may be observed for a hypothesis that α/K has influence over A, these are:

Observation 2: We observe that Positive evidence for a hypothesis that α/K has influence over A is an instance where A is observed following α/K.

Observation 3: We observe that negative evidence for the same hypothesis is an instance where $\neg A$ follows a choice from the $\neg K$ partition.

Observation 4: We observe that counter evidence for the same hypothesis is an instance where $\neg A$ follows α/K.

First, assume that α/K has unambiguous influence over A in a noise free environment. Then, over a number of observations, there would be no instances of counter evidence, a number of instances of positive evidence and at least one instance of negative evidence. The number of instances of negative evidence beyond one is immaterial, it simply serves as a flag that A is not constant. Representing observations of

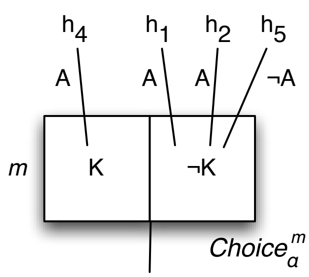


Figure 3. α/K has unambiguous influence

noiseless, unambiguous influence as a branching time choice gives Figure 3 where α/K clearly leads to h_4 where A holds. This satisfies the *strict* reading for both cstit and dstit that was outlined in observation 1. In a noisy environment there may be cases where α/K does not lead to A and such counter evidence would manifest itself as an additional history from the K partition at $Choice^m_\alpha$ where $\neg A$ holds. This means that $Choice^m_\alpha$ no longer satisfies strict stit evaluation rules because α has no choice by which it may guarantee A.

B. Extended influence and gateways

A single agent operating in isolation from other agents may be able to influence its environment in a number of ways. We refer to the set of world states that this agent can bring about by its influence as its *domain of influence* and for this single agent, the *single agent domain of influence* and in a sufficiently complex world this will be a subset of all possible world states.

Definition 3: Given an environment, e, that has a set, E, of possible states and a single agent, α, situated in e then we define *α's* single agent domain of influence as the set of states , $s \in E$, accessible to α as a result its influence, $S^0_\alpha \subsetneq E$.

Here the 0 suffix indicates that we are considering single agent influence with no contribution from other agents. We extend definition 3 to the notion of extended influence by considering the intermediate stage of multiple, independent agents. If an environment contains a number of agents and these agents are independent of each other then we may aggregate their individual influence domains to give a set of reachable states for that world.

Definition 4: Given an environment, e, that has a set, E, of possible states and a set of agents, $G = \{\alpha, \beta, \ldots, \omega\}$, situated in e and with a set of possibly overlapping single agent domains of influence $\{S^0_\alpha, S^0_\beta, \ldots, S^0_\omega\}$. We define $S^0_G = \bigcup_{\alpha \in G} S^0_\alpha$ as the aggregate set of individually reachable states for that group of agents.

The 0 suffix, as in the single agent case, indicates that although we are considering a group of agents we are confining this consideration to single agent influence. Moving on to the two agent case we consider agents α and β situated in an environment where they can influence each other and assume that by working together – either in parallel or by serial cooperation – that at least one agent is able to reach world states that were unreachable to it individually. We must be careful here and emphasise that *is able to reach* means *is able to reach as a result of its action* so that we do not attribute *α's* perceiving a state brought about by β as *α's* reaching that state. This is what we term *extended influence* and we shall define this for two agents before considering a more general definition.

Definition 5: Given an environment, as above, a set of two situated agents, $G = \{\alpha, \beta\}$ and S^0_G, an aggregate set of individually reachable states. We say that S^1_α describes the set of states reachable by α operating in conjunction with one other agent, in this case β. α exhibits extended influence when $S^1_\alpha \backslash S^0_G \neq \emptyset$.

then by working together they may reach world states lying outside of their individual single agent domains. In such cases we state that an agent's individual influence has been *extended* into another domain and, for convenience, refer to these as two agent domains, three agent domains and so on.

Before considering the general case we note that influence domains may be notionally nested relative to a given agent's individual ability. This is illustrated in Figure 4 that may be thought of as representing two agents, α and β "meeting" in their environment, at this point α may give β some token or tool. This is a "gateway action" that allows β to influence a new world state which is contained in the two agent influence domain. It may be that α does not choose to the gateway action and in this case β will remain in the single agent influence domain. If two agents acting in together simultaneously can bring about an individually unreachable state then each agent's influence extends the other agent's ability and does so without either agent *seeing to it* that the other agent does something. Other agent actions, whether

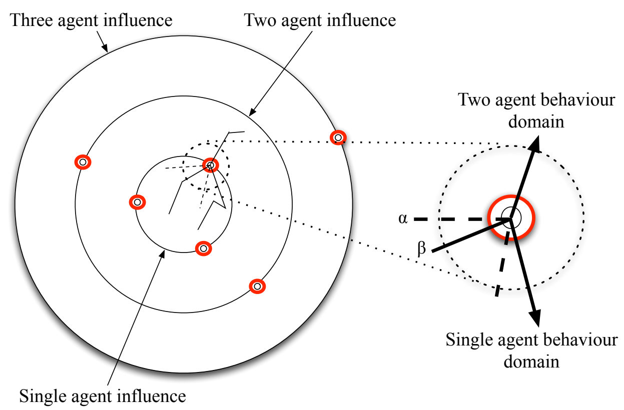


Figure 4. Nested influence domains relative to an agent

these occur simultaneously or in series, that allow an agent to move between domains are characterised as "gateway" actions notionally representing gateways between influence domains, we observe that:

Observation 5: Where one agent, α acts in such a manner as to allow some other agent, β do bring about a world state that was previously inaccessible to β then we observe that α*'s* action is a *gateway* action that allows β to extend its influence into a higher influence domain.

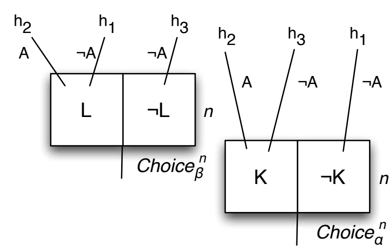
This representation highlights two interesting properties. If agents are to jointly bring about a proposition then the state space must contain that proposition. This may seem like an obvious requirement but it provides a clear representation of the deontic *ought implies can* [19] and one that may be applied to learning and adaptive systems. The second property is that there may be few *gateways*, if a state space is to be searched then locating and analysing gateways will require some form of guided searching.

Returning to our definition of extended influence and bearing in mind the notion of domain nesting we consider the more general case. Nesting brings obvious problems, if two agents acted alternately each extending the other's behaviour causing the world to cycle between individually unattainable states then we may see infinite nesting of repeated states.

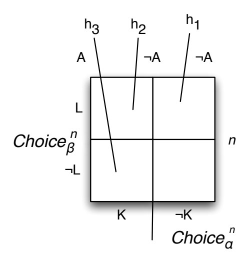
Definition 6: Given an environment, as above, and a set of agents, $G = \{\alpha, \beta, \dots, \omega\}$, situated in e then for a single agent $\alpha \in G$ we define the n^{th} level of nested influence for α as $S^n_\alpha = \bigcup\limits_{i \le n} S^i_\alpha$ and where α has extended influence at level n then $S^n_\alpha \backslash S^{n-1}_\alpha \neq \emptyset$.

Definition 6 allows us to complement levels so as to prevent repeated states from leading to infinite chains. This defines extended influence for a single agent in a world where other agent influence is implicit and may readily be extended to groups of agents as necessary.

Definitions 3 to 6 have viewed influence in a set theoretic manner and we now return to the earlier branching time representation of agent action. The extended influence part of ability may be characterised as a variation in the mapping of histories to the choice partition seen by an agent at a moment. Considering, briefly, the case of a parallel collocated action where two agents act simultaneously and in doing so extend the ability of one or both agents. This may be represented by equations 5 and 6 below. These indicate that a joint action has an effect on the distribution of histories causing it to differ from the distribution in an individual agents choice partitioning. Consider the two agents of Figure 5(a), α and



(a) Individual choice



(b) Parallel choice

Figure 5. Individual and parallel choice

β, with $Choice^m_\alpha = \{K, \neg K\} = \{\{h_2, h_3\}, \{h_1\}\}$ and $Choice^m_\beta = \{L, \neg L\} = \{\{h_1, h_2\}, \{h_3\}\}$. Note that neither α nor $beta$ influences A individually. Only when combined, so that the equivalence classes are further partitioned (see Figure 5(b)) by their joint action is influence evident, as then $Choice^m_{\alpha\|\beta} = \{(L, K), (L, \neg K), (\neg L, K), (\neg L, \neg K)\} = \{\{h_2\}, \{h_1\}, \{h_3\}, \emptyset\}$.

Horty and Belnap [9] use the notation $Choice^m_\alpha(h)$ to represent the particular possible choice by α at moment m that contains history h. This choice may contain other histories and we extend the notation by prefixing it with an h, $h.Choice^m_\alpha(h)$, that reads *some history belonging to the choice partition containing* h. We also combine agents so as to allow a joint choice, $Choice^m_{\alpha\|\beta}$ indicates that the two agents α and β are acting in parallel and exercise their independent choices at moment m.

This allows us to state that in general, a necessary condition for joint action being more influential at a moment m is that:

$$\exists h.Choice^m_{\alpha\|\beta}(h) \subsetneq Choice^m_\alpha(h) \tag{5}$$

or

$$\exists h.Choice^m_{\alpha\|\beta}(h) \subsetneq Choice^m_\beta(h) \tag{6}$$

Informally, joint action by α and β at moment m alters the history or histories passing through some choice partition so that for the joint action the valuations of propositions on histories, the number of histories or both will differ from those of the individual agents making the same choices independently. This allows for a measurement of influence, if the set of world states accessible to α and β acting jointly contains states that are not in the sets of accessible states available to the individual agents. “Measurement” is not linear in that we are considering the number of accessible states, it is based on set membership and we consider the joint reachability of states that are individually unreachable as *more influential.* In this example the equations identify h_6 where A holds. Agent influence in a collocated and

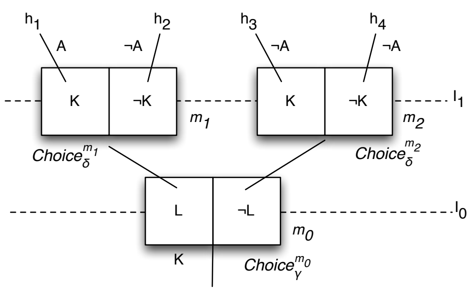


Figure 6. Sequential choice

cooperative action may be thought of as being commutative, α cooperating with β is the same as β cooperating with α. Sequential influence is not commutative and may require an ordering of actions. For example, a β type agent requires some token to allow it to bring about A. If a α type agent acts so as to give the β agent this token then it must do so *before* β chooses its action. This is illustrated by Figure 6 where moments, that are agent choices local to a particular history, potentially occurring at the same time are grouped into sets of *instants*. Here $I_1 = \{m_1, m_2\}$ and $I_0 = \{m_0\}$. If α/L at instant I_1 and this is followed, at instant I_1, by β/K then this guarantees A. Conversely, if $\alpha/\neg L$ I_0 then β has no choice available at $m_2 \in I_1$ that guarantees A. Where we write $Choice^m_{\alpha;\beta}$ we say that β executes an independent choice at m and that this choice has been preceded by a choice by α at some earlier moment. For the sequential case, a necessary condition is that

$$\exists h.Choice^m_{\alpha;\beta}(h) \subsetneq Choice^m_\beta(h) \tag{7}$$

Informally this reads that the mapping of histories for a given choice by β at moment m when preceded by α*'s* action differs from what it would have been in the absence of α*'s* action. Intuitively, α*'s* action plays a part in refining the distribution of histories in β*'s* choice partitioning so as to either remove uncertainty or add new histories and potentially extend β*'s* ability.

IV. Observing and reasoning about influence

In observations 2, 3 and 4 we outlined how evidence presents itself and we note that evidence is countable, a coaching agent may tally how many pieces of evidence it has gathered for each evidence class. The coaching agent has only these data to work with so we use these tallies to generate indices for each hypothesis allowing a coach to rank the hypotheses in its database. The first measure adopted was the ratio between positive and counter evidence and we call this the *P:C* ratio for a hypothesis.

Definition 7: Given a hypothesis that an agent action leads to a proposition holding, a tally of instances of positive evidence for this, *P*, and a tally of counter evidence, *C*, we define the *P:C* ratio for this hypothesis as the ratio of these observed tallies.

Early, and very simple, experiments were carried out with a single class of agents each possessing identical actions:

Definition 8: We define an agent class as a set of agents possessing identical sets of actions.

These experiments indicated that the *P:C* ratio was a useful measure but when noise was introduced or where nested behaviours were being explored the *P:C* ratio alone was not sufficient and we sought another way of using the observed data to rank hypotheses.

Definition 9: Given a hypothesis that an agent action leads to a proposition holding, a tally of instances of positive evidence for this, *P*, and a tally of counter evidence, *C*, we define the *P-C* value for this hypothesis as the difference between the positive evidence tally and the counter evidence tally.

We adopted the *P-C* value as an additional metric allowing coaching agents to filter groups of hypotheses with similar *P:C* ratios. Intuitively a larger *P-C* value for a hypothesis indicates that there are more incidences of it showing influence than not and is, possibly, worth considering synthesising a behaviour based on this hypothesis and distributing it in the agent environment. The negative evidence value was not used here because, as indicated above, it simply acts as an indication that a proposition is changeable, if there was no negative evidence then there would be no hypothesis to consider.

If the environment were noisy and this noise had some effect on the value of A then observations would be unlike those illustrated in Figure 3. Instances of counter evidence may be observed but since observations already indicate that α/K has some influence over A we use the *P:C* ratio (in experiments this was *positive / (positive + counter)* so as to prevent divide by zero errors) to indicate the relative strength of each item of evidence. Clearly if the *P:C* ratio is 1 then influence is unambiguous, if the *P:C* ratio <1 then either the agent has influence and there is some interference or the agent has no influence and something else is changing A. The *P:C* ratio provides a value that allows the ranking of hypotheses and the tracking of the effect of any changes in behaviour. If, for example, α/K has influence over A is contingent on β/L then increasing the incidence of β/L will result in stronger evidence for α/K having influence over A.

If α*'s* influence over A is contingent on a choice by β, either at some earlier time or at the same time, then noisy evidence would be observed but the *P:C* ratio for α*'s* influence will also incorporate the influence of β*'s* action. This other agent influence, when combined with noise, will exhibit a lower *P:C* ratio than unambiguous influence. If α has no influence over A then the *P:C* ratio will be dominated by instances of counter evidence and will be low. This gives three *bands* of evidence, one for unambiguous influence, one for other agent influence and one for no influence. A *P:C* ratio band may contain a number of hypotheses for agent influence but which of these are interesting and worth investigating further? This was investigated by a series of experiments carried out in a simple agent system. In this system a number of reactive *actor* agents of the same type or class were observed by dedicated *coaching* agents. These coaching agents aggregate observations of agent behaviour – an agent's pre-action perception of its environment or *precepts* its action choice and its post-action perception of its environment or *postcepts* – using them both to identify agent influence and as a foundation for synthesising behaviours that will maximise agent influence individually and collectively.

A. Finding unambiguous influence in a noisy environment

Coaching agents were able to observe actor agents by examining history data that actors left in the environment. Based on these observed data coaching agents generated hypotheses for actor agent influence and seed the environment with new behaviour patterns based on what were considered good hypotheses. Identifying good hypotheses solely by observation presented some interesting problems. The notion was that if coaching agent hypotheses correctly identified agent influence and biased agents towards exercising that influence then this would be reflected in later observations.

Simple experiments in an extremely noisy environment demonstrated that ranked *P:C* ratios tended to fall into bands [4]. Unambiguous agent influence hypotheses were grouped at the top with hypotheses containing influence that was potentially contingent on other agents falling into a group separated from and below the top hypotheses. A third band containing "poor" hypotheses separated out at the bottom of the ranking. The banding of *P:C* ratios is, intuitively, heavily dependent on factors such as environmental noise and, in cases of other agent influence, the number of instances of β selecting the appropriate action to enable α to bring about A. The banding of hypotheses indicated that unambiguous influence was easily detected but ranking by *P:C* ratio alone was insufficient to allow the identification of potential other agent candidate hypotheses.

B. Finding other agent influence in a noisy environment

At this point evidence for hypotheses is considered solely on the basis of metrics, described above, generated from positive and counter evidence tallies. The *P:C* ratio alone proved to be insufficient and although considering it in conjunction with the *P-C* value yielded some improvement but a noisy, multi agent environment still presented difficulties. Further experiments led to a system of prefixing the hypotheses with agent precepts. This allows an agent to use what it perceives of its world as a filter to refine how it may use its ability. The intuition behind this is twofold, firstly the hypotheses would embody the notion that, neglecting effects of noise and other agents, an agent's ability is truly contingent on its current state and the states that are reachable from that state. The second intuition was that this would provide finer means of filtering hypotheses. Rather than a general hypothesis that α choosing action K leads to A holding, which we write as $\alpha/K \rightsquigarrow A$, with a single set of evidence tallies a hypothesis is grouped with a set of precepts, P, then α*'s* choosing K leads to A and we write this $\alpha^P/K \rightsquigarrow A$. Using precepts as a filter allows the introduction of preconditions for an agent's ability and in doing so we gain the means to analyse sequences of actions. Recalling the deontic notion briefly mentioned in Section III-B, this precept filtering gives a *situationist* dimension, as described by Hansson [22], which allows for agent ability to be categorised by the agent's situation. Coaching agents are thus able to select the most influential hypothesis for a given set of agent precepts. These explorations of simple systems, in conjunction with work on a partial logical characterisation of influence operators, provided sufficient information to build a coaching agent for use in a simple world.

C. Experimental observations

The single agent class experiments, outlined in Section IV, were extended to use two agent classes and provide a system that would test the practical application of the influence theory outlined above. These were based on the idea of incrementing an accumulator and brought the possibility of agents being able to extend their influence by repeated actions. Actor agents were, initially, simple stochastic agents with a fixed set of actions from which they would choose at random, each of the choices was weighted evenly so as to give a flat probability distribution. Coaching agents would be able to observe agent influence but would be unable to see that an influence was bringing the system closer to a specific value that was considered as being a goal state. When coaching agents observed influence they generated new behaviours that are probability distributions biased towards a particular influential agent choice. The use of a numerical accumulator is similar to the notion of bridge building, agents bridging a gap may exhibit extended influence by their being able to move further across a gap but the results of that extended influence will not be evident until the gap has been successfully bridged.

Initial validation experiments with a single agent class showed that both in noiseless and noisy settings agent coaching led to agents behaviour rapidly becoming biased towards incrementing the accumulator. The system was extended to two agent classes with one class, α being able to increment an accumulator from zero and the other class, β only able to increment non zero accumulators. This provided a simple case of nested other agent ability, $[\alpha\ influences\colon [\beta\ influences\colon A]]$. It was observed that both agent classes were coached so as to maximise their influence and the environment was extended so as to contain a number of agents of each class and a number of accumulators. Coaching, again, had agents rapidly maximising their influence and as agents moved to other accumulators they were quickly able to increment accumulators to target values. An interesting point in this experiment is that "good" aggregate behaviour patterns involve very little observable action from α class agents. The number of instances of α class agents incrementing an accumulator from its zero state will be small when compared with the number of instances of β class agent influence resulting from repeated incrementation. The frequent influence of β class agents ought to be easy to spot but what of the relatively small contribution of α class agents? This is a contribution that is so small that it may be readily swamped in a noisy environment. Evidence is based on observation and in a noisy or uncertain environment and, as noted in Section III-A, we may expect to observe some counter evidence to an agent's ability. Observed counter evidence does not necessarily mean that an agent has no influence. It may be that $\alpha's$ ability is contingent on another agent or it may be that another agent acted simultaneously so as to counter $\alpha's$ choice. Experimental data from an agent test are presented in Table I. The first column, ID, is simply an ID tag for a particular hypothesis. The precepts column contains data from an agent's precepts and the two digits indicate the presence of an accumulator and if the accumulator is zero respectively. The third column describes the hypothesis in question, for example, $\beta/5 \rightsquigarrow AC : 3$ says that the hypothesis is that a β class agent choosing action 5

Table I
HYPOTHESIS DATA, NOISY ENVIRONMENT

ID	Precepts	Hypothesis	P:C	P-C
Generated behaviours				
0	10	$\beta/5 \rightsquigarrow AC : 3$	0.884134	13122
8	10	$\alpha/4 \rightsquigarrow AC : 3$	0.627907	11
1	11	$\alpha/5 \rightsquigarrow AC : 1$	0.00050045	-9981
Hypothesis database, P:C ordering				
0	10	$\beta/5 \rightsquigarrow AC : 3$	0.884134	13122
8	10	$\alpha/4 \rightsquigarrow AC : 3$	0.627907	11
6	10	$\alpha/6 \rightsquigarrow AC : 3$	0.580645	5
9	10	$\alpha/2 \rightsquigarrow AC : 3$	0.555556	4
10	10	$\alpha/0 \rightsquigarrow AC : 3$	0.53125	2
3	10	$\alpha/3 \rightsquigarrow AC : 3$	0.0597125	-10780
Hypothesis database, P-C ordering				
0	10	$\beta/5 \rightsquigarrow AC : 3$	0.884134	13122
8	10	$\alpha/4 \rightsquigarrow AC : 3$	0.627907	11
6	10	$\alpha/6 \rightsquigarrow AC : 3$	0.580645	5
9	10	$\alpha/2 \rightsquigarrow AC : 3$	0.555556	4
10	10	$\alpha/0 \rightsquigarrow AC : 3$	0.53125	2
7	10	$\beta/1 \rightsquigarrow AC : 3$	0.00885609	-1331

leads to transition type 3 being observed in an accumulator. Type 3 transitions indicate that the accumulator incremented and type 1 transitions indicate that the accumulator became non zero. The remaining columns list the *P:C* and *P-C* values for observations of these hypotheses. Since the coaching agent had no cognitive or reasoning abilities, its operation was based solely on observation, generation of hypotheses and seeding of hypotheses selected by the *P:C* and *P-C* metrics. Table I is divided into three horizontal groups, the top group lists the hypotheses chosen as seeds for generated agent behaviours. These are the most influential behaviours for each agent class and each transition or change in the environment. Behaviour ID 1, $\alpha/5 \rightsquigarrow AC : 1$ has very poor *P:C* and *P-C* values but it remains the most influential behaviour for α agents leading to transition 1 and because of this it is considered suitable for seeding. The groups below show the top six hypotheses in the database ordered by *P:C* and *P-C* values.

Our intuitive knowledge of the problem indicates that α class agents have an influential action – the tipping of an accumulator from zero to some non zero value – and that the occurrence of this action would be very small. The *P:C* and *P-C* orderings worked as expected, the most influential actions rose to the top of the ordered tables. The *P-C* value is intended as a filter to offer further differentiation of hypotheses with very similar *P:C* ratios. However, the gateway action, α's initialisation of the accumulator to a non zero value, does not appear in either ordering. Earlier work had led us to consider the nature of hypotheses which, until then, had been global, that is they took no account of the agent's immediate environment and were based solely on an agent choice or action. This meant that coaching agents had an implicit assumption that, for example, a hypothesis that $\beta/5 \rightsquigarrow AC : 3$ would be applicable in all circumstances. This is clearly far too coarse, the transition from zero to non zero is only going to be occur if the accumulator is zero before the agent makes its choice of action. This led us to generate "prefixed" hypotheses. The prefixes were, as described above, the agent's pre-choice precepts. Returning to the generated behaviours section of Table I and using precepts we see that given precepts 10 (the presence of a non zero accumulator) the hypothesis $\beta/5 \rightsquigarrow AC : 3$ exhibits the greatest influence for β class agents.

The most influential hypothesis for α class agents given the same precepts is $\alpha/4 \rightsquigarrow AC : 3$. This illustrates the overwhelming effects of noise generated by the actions of other agents. α class agents have no influence here but if an α class agent is collocated with a β class agent and the β agent does increment the accumulator then that change will appear in the α class agent's postcepts whatever action it selects. The hypothesis ranking shows that the β class behaviour is clearly more influential and this is, indeed, correct. For a different set of precepts, the presence of a zero accumulator, we have only observed a small number of influential actions but for this set of precepts it is the most influential action observed so is a candidate for seeding a new behaviour for α class agents.

Agents were built so as to accommodate multiple behaviours. These are represented as sets of weightings for each of the agent's possible choices or actions. The default behaviour has a flat weighting making each of its actions equally likely by random selection. Acquired behaviours are biased towards a particular action and the strength of this bias depends on the strength of the coaching agent's evidence for that action being influential. Each acquired behaviour has an associated percept pattern that is used to select the appropriate bias to apply to action selection.

We noted, above, that agents have a fixed fixed set of abilities, they are unable to acquire new choices or actions but are able to acquire new preferences or weightings for the selection of these abilities. Combining these fixed abilities with precept prefixed hypotheses allows us to consider actor agents as a mapping of precepts on to preferred behaviour patterns. This is illustrated by the schematic agent architecture of Figure 7 where an agent consists of a collection of behaviours selected by some precept filtering mechanism. Conceptually actor agents are a collection of finite state machines each with a weighted stochastic transition selection mechanism.

The agent, illustrated in Figure 7, has six choices available to it, $Choice_{agent} = \{K, L, M, N, O, P\}$. Each of these choices represent some action available to the agent, the choice of action is individual and independent of other agents but it may be in concert with other agents or it may be a null action. The set of choices remains constant throughout an agent's life, it cannot acquire new choices

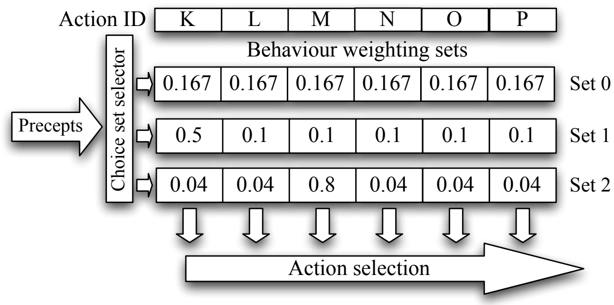


Figure 7. Agent internals – behaviour stack holding three selectable behaviour patterns for actions K to P.

and it cannot discard any of its current set of choices. The agent's choice set is overlaid by a set of weightings with each weighting assigning a preference for a single choice. Choice set 0, in Figure 7, gives each of the elements of $Choices$ an equal weighting, this is the agent's default state and it has no individual behaviour characteristics, each of its actions has an equal chance of being chosen. Behaviours 1 and 2 are acquired behaviours that have been synthesised by a coaching agent. Behaviour 1 is biased towards action K and behaviour 2 has a stronger bias towards action M. A behaviour pattern will not be allowed to become an absolute choice, we maintain a very small stochastic element so as to prevent the system becoming trapped by false hypotheses and to allow agents to adapt to changes in their environment. When the actor agent is operating it uses its precepts to select an appropriate behaviour pattern, it there is no match then it selects the default pattern It then generates a random number, checks this against the weightings in the selected choice set and executes the its choice.

Coaching agents generate a database of observations during their operation. This database contains hypothesis data, as listed in Table I, and is structured so as to reflect hypotheses overlaid on branching time. The notion of branching time brings difficulties, it is unbounded in nature and its relentless forwards branching is clearly not suited to bounded computational systems. To address this difficulty we admit the use of loops in the branching time structure. Where an agent's action leads to a particular state then we link that action to a "state bucket" that contains a chain of hypotheses for agent actions given that state as a precept. If a hypothesis leads to the same state then the link from that hypothesis goes to the state bucket at the root of its chain. If there are multiple versions of the same $agent/action$ hypothesis in a chain each will lead to a unique next state. This brings the database structure closer to the notion of *leads to* and *may lead to* relations that are embodied in our hypotheses and discussed in some detail in Section V. Figure 8 illustrates this, the boxes on the left hand side are "state

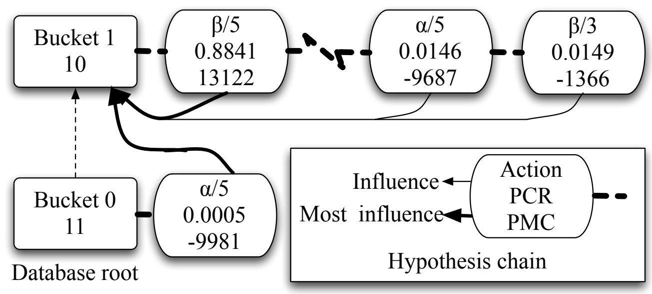


Figure 8. Partial coaching hypothesis database from experimental data

buckets" that represent the agent precept prefixes described in Section IV-B. Precepts are shown as two binary flags that indicate presence and state of an accumulator. Bucket 1 holds a list of influential actions when agents perceive a non zero accumulator and bucket 0 holds a list of influential actions where agents perceived a zero accumulator. The dotted lines leading right from the state buckets indicate hypothesis chains, from bucket 1 it can be seen that $\beta/5$ leads to a non zero accumulator (influence is identified by a change in accumulator value) with a *P:C* ratio of 0.8841 and a *P-C* value of 13122. Further along that chain we see that $\alpha/5$ leads to a non zero accumulator with a *P:C* ratio of 0.0146 and a *P-C* value of -9687. The most influential hypotheses in a chain are represented by a heavier line from the hypothesis entry to the finishing state bucket.

By inspecting the data it may be seen that from state 11 there is no influential action available to β class agents and that, despite the very low *P:C* ratio and poor *P-C* value, the $\alpha/5$ hypothesis is the most influential that we have observed from a world state where the accumulator is zero. In inspecting the data of Figure 8 we apply human reasoning and infer that if a system is in state 11 then the only way to activate β class agent influence is for α class agents to take the system out of state 11. A simple coaching agent only "sees" that there are some actions that, from observed evidence, are more likely to be influential than others and it seeds agents with these behaviours. The system may behave as intended but does so as a result of random interaction between independent behaviours, the coaching operation has simply increased the incidence of influential behaviours. In order to generate patterns of behaviour that cause agents to behave as required the coaching agent needs some means to reason about individual behaviours in a way that allows for sequencing, combination and substitution. This may also allow the coaching system to coordinate agent behaviours, perhaps by the use of null actions that Halbwach [23] indicates is standard practice in synchronous reactive systems programming. In order to do this we need to characterise agent influence logically using a framework as close to that of standard stit as possible. This will indicate what a coaching agent may safely infer from observed sequences of behaviour and guide how it may synthesise more complex aggregate behaviours. In the following section we outline some aspects of this work.

V. Logical aspects of influence

In order to explore logical properties of influence we investigate the *leads to* operator introduced in Section IV-B. Our theory is based on observations that may provide evidence of influence. The first, $\rightsquigarrow$ a *leads to* operator, may be considered as being similar to a stit operator. For $\rightsquigarrow$ to hold we must observe evidence of ability and consistent results. Although we think of the *leads to* as stit-like it differs from stit in that it is based entirely on observation and does not necessarily represent a complete characterisation of an agent's choices or exploration of its world. The similarity allows the characterisation of influence in a modal setting, the $\rightsquigarrow$ operator is very close to standard stit and in an ideal setting may be equivalent. If an agent, β*'s*, ability is contingent, perhaps on another agent's choice that must occur before or at the same moment as β*'s* choice then we may see inconsistent evidence for β*'s* ability. To allow for this we introduce a *may lead to* operator that we write as $\Diamond\!\!\rightsquigarrow$. This has weaker evidence requirements, a hypothesis based on the notion that an agent choice *may lead to* S allows for counter evidence. The presence of counter evidence is taken as an indication that the agent's ability is not fully understood and may be dependent on other agents or environment factors. Given an agent, β with a choice K and a proposition A if we observe instances of $\beta/K \rightsquigarrow A$ and instances of $\beta/K \not\rightsquigarrow A$ we infer that β/K may lead to A and write this as $\beta/K \Diamond\!\!\rightsquigarrow A$.

In addition to these two operators we extend standard modal logic by the introduction of an *other-agent* extension. In the standard, single agent, versions of modal rules and axioms there are a number of sentences that may admit multiple agents. The standard modal axiom C, for example, using $\rightsquigarrow$ in place of stit, may be written as equation 8 and multi agent extensions may be considered.

$$C. \quad [\alpha \rightsquigarrow : A] \wedge [\alpha \rightsquigarrow : B] \supset [\alpha \rightsquigarrow : A \wedge B] \qquad (8)$$

The standard single agent / multiple proposition statement becomes a multiple agent / single proposition statement. Casting C in this mould replacing the propositions, A and B with agents α and β and having those agents act sequentially on a single proposition gives C_{agent} which is written as equation 9.

$$C_{agent}. \quad [\alpha \rightsquigarrow : A] \wedge [\beta \rightsquigarrow : A] \supset [\alpha ; \beta \rightsquigarrow : A] \qquad (9)$$

These agent extensions must be considered both as parallel cases, outlined above, and as serial cases. C_{agent}, in serial form states that $[\alpha \rightsquigarrow : A] \wedge [\beta \rightsquigarrow : A] \supset [\alpha ; \beta \rightsquigarrow : A]$. Where C_{agent} holds it appears to indicate that α and β*'s* actions are not mutually exclusive.

Because the theory is based on observed evidence and is being investigated in a noisy setting, standard modal logic rules and axioms may not be falsified as cleanly as with strict stit. Noise may cause counter evidence to appear and this would cause a strict stit reading to fail. We examine two cases, the convergence axiom C and the rule of equivalence RE, chosen from a larger set of modal rules and axioms, as examples for a partial characterisation. These demonstrate the other agent extension in two very different settings. Taking C and C_{agent} as examples agents may generate the data of Tables II and III. Note that in the following tables N represents negative evidence as described in observation 3. This is represented by a tick because, as indicated in Section III-A we require only that we observe at least one instance of negative evidence, this is not tallied as we do for positive evidence, P, and counter evidence, C.

A. *Influence and the convergence axiom, C*

If an agent were to see to it that A holds and that B holds at an instant and its choice of action is the coincidental result of its having two independently reasoned goals then it does so without intending to see to it that A and B hold jointly. Neglecting the agent's intent, however, it would be difficult to deny that the agent does see to it that A and B do hold jointly and that the principle stated in equation 8 is supported. A noteworthy point here is that operators are constrained to an agent and choice pair, to say that $\alpha/K \rightsquigarrow A$ and $\alpha/K \rightsquigarrow B$ implies is that it is the same choice, K, that brings about both A and B. Assuming that equation 8 holds an attempt to generate counter examples is illustrated in Table II.

Table II
EVIDENCE SUPPORTING C

Hypothesis	Evidence P	Evidence N	Evidence C	P:C ratio	Conclusion
(IIa): $\rightsquigarrow$ example					
$\alpha/K \rightsquigarrow A$	n	✓	0	∞	$\alpha/K \rightsquigarrow A$
$\alpha/K \rightsquigarrow B$	n	✓	0	∞	$\alpha/K \rightsquigarrow B$
$\alpha/K \rightsquigarrow A \wedge B$	n	✓	0	∞	$\alpha/K \rightsquigarrow A \wedge B$
(IIb): $\Diamond\!\!\rightsquigarrow$ example					
$\alpha/K \rightsquigarrow A$	n	✓	p	n/p	$\alpha/K \Diamond\!\!\rightsquigarrow A$
$\alpha/K \rightsquigarrow B$	q	✓	r	q/r	$\alpha/K \Diamond\!\!\rightsquigarrow B$
$\alpha/K \rightsquigarrow A \wedge B$	s	✓	t	s/t	$\alpha/K \Diamond\!\!\rightsquigarrow A \wedge B$

In the noiseless case of Table IIa it can be seen that the hypotheses match the conclusion because of the lack of noise induced counter evidence, that is $C = 0$. It the noisy case of IIb, where $C > 0$, we apply the may lead to operator to the conclusion.

B. *The convergence axiom with agent extension, C_{agent}*

C_{agent} is the result of applying the other agent extension, as outlined above, to the standard C axiom and represents scenarios where a number of agents with potentially similar abilities act either in parallel or in series. This gives two versions, equation 10 says that if α can see to it that A and if β can see to it that A then α and β acting simultaneously can see to it that A holds. Equation 11 says that if α can see to it that A and if β can see to it that A then α and β acting serially can see to it that A holds.

$$C_{\|agent}.\ [\alpha \rightsquigarrow: A] \| [\beta \rightsquigarrow: A] \rightarrow [\alpha \| \beta \rightsquigarrow: A] \quad (10)$$

$$C_{;agent}.\ [\alpha \rightsquigarrow: A] \wedge [\beta \rightsquigarrow: A] \rightarrow [\alpha; \beta \rightsquigarrow: A] \quad (11)$$

$C_{;agent}$ is a potentially dangerous property. Consider agent actions that are mutually exclusive, for example, A may represent "pick up an indivisible token". If two agents, α and β have an action that may bring about A. Because coaching agent observations are limited to changes in the environment we must be careful that when both agents act so as to bring about A and although A may result α*'s* influence and β*'s* influence are mutually exclusive. The potential danger here is that a coach may identify this as a joint action when only one agent was responsible. Considering the parallel action case first, if α and β simultaneously act so as to bring about A then A will hold but it will do so as a result of α*'s* action or β*'s* action and not as a result of both actions. It is assumed that both agents have different abilities, that is that they execute different choices but these choices are functionally equivalent as far as A is concerned. If, say, $\alpha/K \rightsquigarrow A$ and $\beta/L \rightsquigarrow A$ and $\neg A$ holds then when agents act simultaneously both will perceive that A holds after their action. The data of Table III illustrates potential

Table III
EVIDENCE AND PARALLEL C_{agent}

Hypothesis	Evidence P	Evidence N	Evidence C	P:C ratio	Conclusion
$\alpha/K \rightsquigarrow A$	n	✓	p	n/p	$\alpha/K \Diamond\!\!\rightsquigarrow A$
$\beta/L \rightsquigarrow A$	q	✓	r	q/r	$\beta/L \Diamond\!\!\rightsquigarrow A$
$\{\alpha/K \| \beta/L\} \rightsquigarrow A$	s	✓	t	s/t	$\{\alpha/K \| \beta/L\} \Diamond\!\!\rightsquigarrow A$

observed evidence for C_{agent}. Even in a single cell world where agents are always collocated the number of instances of $\alpha/K \rightsquigarrow A$ and $\beta/L \rightsquigarrow A$ will be greater than those of $\{\alpha/K \| \beta/L\} \rightsquigarrow A$. Assuming an even distribution of noise the *P:C* ratios n/p and q/r will be greater then s/t and ranking these as per the discussion in Section IV makes the single agent hypotheses appear to be more influential than the parallel action hypothesis. Returning to agents, there are two hypotheses for each – a single agent hypothesis which indicates an ability to bring about A and an other agent hypothesis indicating the same. Returning to the notion of gateways between domains of influence illustrated in Figure 4 it can be seen that the single agent hypothesis is contained in the single agent influence domain which is, by extension, contained in the two agent influence domain. After ranking, the simpler single agent hypotheses are seen to carry more influence and offer a better account of α and β's individual ability to influence A than the two agent hypothesis.

If α and β operate sequentially then coaching agents will see much more evidence of influence for single agent action than for serial action. Given $\alpha/K \rightsquigarrow A$ immediately followed by $\beta/L \rightsquigarrow A$ the latter action will, except when noise intervenes, show no influence as A already holds and no change will be evident following β/L.

Whilst C_{agent} is not strictly falsified it may be seen that the foundations of the influence theory – observations of agent behaviour – do not lead to circumstances where C_{agent} may be considered as valid.

C. Rule of equivalence RE

Chellas [24] lists the rule of equivalence as:

$$RE. \ \frac{A \leftrightarrow B}{\Box A \leftrightarrow \Box B} \tag{12}$$

Casting this into influence based view gives:

$$RE. \ \frac{A \leftrightarrow B}{[\alpha \rightsquigarrow: A] \leftrightarrow [\alpha \rightsquigarrow: B]} \tag{13}$$

RE says that equivalent propositions are equally necessary. This relates to several aspects of agent behaviour in a coached environment. We must be careful with the notion of equivalence. If two propositions are absolutely equivalent, that is to say that they are the same but simply carry different labels or names, then one would expect to see evidence tallies matching exactly even in a noisy multi agent environment. If, however, the equivalence is both propositions are the result of the same choice then evidence tallies may not match exactly. In this case a coach would observe evidence supporting two hypotheses, one that a given action leads to A and another that the same action leads to B.

Let us assume two hypotheses, one that $[\alpha \rightsquigarrow: A]$ and one that $\neg[\alpha \rightsquigarrow: B]$ for α/K. Let us also assume that α/K brings about A and brings about B

Table IV
EVIDENCE SUPPORTING RE

Hypothesis	Evidence P	Evidence N	Evidence C	P:C ratio	Conclusion
(IVa): $\rightsquigarrow$ example					
$\alpha/K \rightsquigarrow A$	n	✓	0	∞	$\alpha/K \rightsquigarrow A$
$\neg\alpha/K \rightsquigarrow B$	p	✓	q	p/n	$\neg(\neg\alpha/K \rightsquigarrow B)$
$\alpha/K \rightsquigarrow B$	n	✓	0	∞	$\alpha/K \rightsquigarrow B$
(IVb): $\Diamond\!\!\rightsquigarrow$ example					
$\alpha/K \rightsquigarrow A$	n	✓	p	n/p	$\alpha/K \Diamond\!\!\rightsquigarrow A$
$\neg\alpha/K \rightsquigarrow B$	q	✓	r	q/r	$\neg(\neg\alpha/K \Diamond\!\!\rightsquigarrow B)$
$\alpha/K \rightsquigarrow B$	s	✓	t	s/t	$\alpha/K \Diamond\!\!\rightsquigarrow B$

In the noiseless example of Table IVa we see that the equivalence of A and B is reflected in the positive, negative and counter evidence tallies. Assuming that the $\alpha/K \rightsquigarrow A$ hypothesis holds, as in the statement above, and that A and B are equivalent propositions then there will be counter evidence for the $\neg\alpha/K \rightsquigarrow B$ hypothesis. The conclusion for the negative hypothesis, $\neg\alpha/K \rightsquigarrow B$, is that it does not hold.

The counter evidence that negates the $\neg\alpha/K \rightsquigarrow B$ hypothesis is evidence of a noisy environment where we consider a may lead to result. Here the P:C ratio for $\neg\alpha/K \rightsquigarrow B$ will be significantly smaller than that for both $\alpha/K \rightsquigarrow A$ and $\alpha/K \rightsquigarrow B$. Note that n/p and s/t are not necessarily equal, other agents may play a part in generating observed evidence.

RE considers equivalent propositions from the point of view of a single actor as the agent of change. In a multi agent environment the ability to extend a hypothesis across groups of agents, agents which are members of some equivalence class, will allow a coach to develop behaviours applicable to a greater number of actor agents.

D. The rule of equivalence with agent extension RE_{agent}

Before considering the agent extension to RE we address the question of *agent equivalence*, outlined above, which allows a coach to infer that one agent's ability to bring about A is transferable to other agents. Agent ability is driven by agent choice and we call similarly capable agents *choice class equivalent*. Given two agents, α and β, with choice sets $Choice_\alpha$ and $Choice_\beta$ respectively we say that α and β are choice class equivalent for a choice K iff $K \in \{Choice_\alpha \cap Choice_\beta\}$. Agents belonging to the same agent class will be choice class equivalent by default and the definition above may be extended across agent classes where these classes share common agent choices. This refinement of agent class equivalence to choice class equivalence removes a degree of coarseness from coach reasoning.

RE_{agent}, equation 14, maps RE's claim of the equivalence of propositions onto the domain of agents and allows the coaching operation to work with equivalent agent classes and to substitute equivalent actions in synthesised behaviours. The coaching operation treats agents as abstract entities, they are simply a collection of choices. It may be that different coaching agents use different representations of the same agent class, these representations may simply be different orderings of agent choices so that α/K and β/L amount to the same choice. At an agent class level RE_{agent} allows the coaching operation to aggregate data from several coaching agents each of which may have a different ordering on agent choice sets. Two agents from different classes may have equivalent choices, these may be the same choice contained in different agent choice sets or they may be different choices that lead to the same result. In such cases the coaching operation may substitute agent classes in a synthesised behaviour and this means that it may be possible for more sophisticated coaching agents to be able to optimise synthesised behaviours for agent populations.

$$RE_{agent}. \quad \frac{\alpha/K \leftrightarrow \beta/L}{[\alpha \rightsquigarrow: A] \leftrightarrow [\beta \rightsquigarrow: A]} \tag{14}$$

This is an important rule for our system, if RE_{agent} fails then a coaching agent can not assume that a good behaviour exhibited by one agent of a particular choice class may be transferred successfully to another agent of that class.

Table V
EVIDENCE SUPPORTING RE_{agent}

Hypothesis	Evidence			P:C ratio	Conclusion
	P	N	C		
$\alpha/K \rightsquigarrow A$	n	✓	p	n/p	$\alpha/K\Diamond\!\!\rightsquigarrow A$
$\beta/L \rightsquigarrow A$	q	✓	r	q/r	$\beta/L\Diamond\!\!\rightsquigarrow A$

Given two agents, α and β, belonging to separate agent classes and two different actions, K and L for α and β respectively, a coach may see evidence such as that of Table V. Both α and β present evidence of being able to bring about A and in this case the coach will simply consider this an equivalent ability for each agent class and seed a behaviour for each class. Functionally both behaviours are equivalent, the coach is unable to see a difference and simply treats them as equivalent.

VI. CONCLUSION AND FURTHER WORK

The motivation for this work was to investigate how agents may influence each other and to apply this concept of influence to an analysis of nested other agent behaviour. The investigation grew from earlier attempts at characterising emergent behaviour in simple systems of reactive agents, Logie et al. [3], which proved to to be a difficult problem and led to the consideration of deontic logics. This work is a step towards addressing that problem with new and more suitable tools.

The database structure of Figure 8 followed investigations into data mining sets of coaching agent observations, Logie et al. [2] and the realisation that branching may be collapsed into a potentially closed structure. The data of Figure 8 and Table I were generated by coaching agents observing actor behaviour in a noisy environment and one where only tiny amounts of influential α class agent behaviour would be evident. Given the noise and small incidence of influential α class behaviour these are impressive results (but always with the proviso that they are from a simple system).

Our partial characterisation indicates that, in common with stit, influence supports modal operators. More importantly the characterisation indicates that our theory of influence may be extended into domains requiring complex sequences of agent actions. The failed characterisations are equally important, the other agent extension indicated that our notion of influence rejects cases where agent actions may be mutually exclusive. These results bode well for further investigation to build a more solid understanding of exactly how a coach may manipulate evidence of uncertain ability without generating unrealistic conclusions.

This work, the theory of influence, implementation in a simple test system and a partial characterisation, is the first step in developing tools to investigate adaptive and emergent behaviour in systems of simple agents.

Future work will involve further logical analysis of influence by way of *leads to* and *may lead to* operators. We inferred nested influence from coaching agent database structure in this simple case but are unsure of what other inferences may be safely made. Further investigation into the logical properties of these operators will indicate what inferences may and may not be safe and this will allow experiments with richer and more complex environments where nested influence is not so obvious. Halbwach [23] indicated that the use of null actions was standard in programming synchronous reactive systems. This is something that needs to be investigated and has many interesting threads, how may a coaching agent detect that synchronisation can improve an aggregate behaviour? When and how do agents use synchronising actions and do they need more refined percepts to, perhaps, detect other agents?

Application of our influence theory in complex problem domains are being explored, notably software maintenance where the influence of a developer on a type or class of problem may be observed by a third party allowing for a recommendation system that works by observing existing defect and update management systems.

REFERENCES

[1] R. Logie, J. Hall, and K. Waugh, "Agent influence and nested other-agent behaviour," in *Proc. Future Computing, Service Computation, Cognitive, Adaptive, Content, Patterns, 2009. COMPUTATIONWORLD '09. Computation World:*, November 2009, pp. 488 –493.

[2] R. Logie, J. G. Hall, and K. G. Waugh, "Towards mining for influence in a multi agent environment," in *IADIS European Conference on Data Mining 2008 (part of MCCSIS 2008)*, A. P. Abraham, Ed. IADIS, 2008, pp. 97–101.

[3] R. Logie, J. G. Hall, and K. G. Waugh, "Reactive food gathering," in *CLIMA VI*, ser. Lecture Notes in Computer Science, F. Toni and P. Torroni, Eds., vol. 3900. Springer, 2005, pp. 406–413.

[4] R. Logie, "A study of agent influence in nested agent interactions," Ph.D. dissertation, The Open University, June 2009.

[5] R. A. Brooks, *Cambrian intelligence: the early history of the new AI*. Cambridge, MA, USA: MIT Press, 1999.

[6] N. Jennings, K. Sycara, and M. Wooldridge, "A roadmap of agent research and development," *Autonomous Agents and Multi-Agent Systems*, vol. 1, no. 1, 1998, pp. 7–38.

[7] N. D. Belnap, Jr. and M. Perloff, “Seeing to it that: a canonical form for agentives,” *Theoria*, vol. 54, 1988, pp. 175–199.

[8] B. F. Chellas, *The Logical Form of Imperatives*. Stanford, California: Perry Lane Press, 1969.

[9] J. F. Horty and N. D. Belnap, Jr., “The deliberative STIT: A study of action, omission, and obligation,” *Journal of Philosophical Logic*, vol. 24, no. 6, 1995, pp. 583–644.

[10] A. N. Prior, *Past, Present and Future*. Oxford: Oxford University Press, 1967.

[11] R. H. Thomason, “Indeterminist time and truth-value gaps,” *Theoria*, vol. 36, 1970, pp. 246–281.

[12] R. H. Thomason, “Combinations of tense and modality,” in *Handbook of Philosophical Logic, Volume II: Extensions of Classical Logic*, D. Gabbay and F. Günthner, Eds. Dordrecht: D. Reidel Publishing Co., 1984, pp. 135–165.

[13] B. F. Chellas, “Time and modality in the logic of agency,” *Studia Logica*, vol. 51, 1992, pp. 485–517.

[14] N. Belnap, M. Perloff, and M. Xu, *Facing the Future: Agents and Choices in Our Indeterminist World*. Oxford: Oxford University Press, 2001.

[15] J. Ferber and J.-P. Müller, “Influences and reaction: a model of situated multiagent systems,” in *In: Second International Conference on Multi-agent Systems*. AAAI Press, 1996, pp. 72–79.

[16] F. Michel, “The IRM4S model: the influence/reaction principle for multiagent based simulation,” in *AAMAS ’07: Proceedings of the 6th international joint conference on Autonomous agents and multiagent systems*. New York, NY, USA: ACM Press, 2007, pp. 1–3.

[17] D. Weyns and T. Holvoet, “A model for situated multi-agent systems with regional synchronization,” in *10th International Conference on Concurrent Engineering, Agent and Multi-Agent Systems, CE’03*. Springer-Verlag, 2003, pp. 177–188.

[18] O. Simonin, A. Lanoix, S. Colin, A. Scheuer, and F. Charpillet, “Generic expression in B of the influence/reaction model: Specifying and verifying situated multi-agent systems,” INRIA, Lorraine, Tech. Rep. RR-6304, 2007.

[19] J. F. Horty, *Agency and deontic logic*. Oxford University Press, 2001.

[20] L. Åqvist and P. Mullock, *Causing Harm*, R. Posner and G. Megle, Eds. de Gruyter, 1989.

[21] R. Milner, *Communication and concurrency*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1989.

[22] S. O. Hansson, “Situationist deontic logic.” *Journal of Philosophical Logic*, vol. 26, no. 4, 1997, pp. 423–448.

[23] N. Halbwachs, *Synchronous programming of reactive systems*. Kluwer Academic Publishers, 1993.

[24] B. F. Chellas, *Modal Logic, an introduction*. Cambridge University Press, 1980.

Adaptation and Self-adaptation of Developmental Multi-Robot Systems

Serge Kernbach, Eugen Meister, Florian Schlachter, Olga Kernbach
Institute of Parallel and Distributed Systems, University of Stuttgart, Germany
Email: {Serge.Kernbach, Eugen.Meister, Florian.Schlachter, Olga.Kernbach}@ipvs.uni-stuttgart.de

***Abstract*—In this work we explore several adaptive and self-adaptive processes in systems with a high degree of developmental plasticity. It is indicated that such systems are driven by two different forces: design goals and self-concept, which define what the system "should be" and "may be". The paper discusses mechanisms, leading to adaptive and self-adaptive behavior, as well as possible conflicts between them. Bound and unbound self-concepts are introduced. The discussed mechanisms are exemplified by a collective locomotion of reconfigurable multi-robot system, where several self-organizing and evolutionary approaches are exploited.**

***Keywords*-Collective robotics, artificial multi-robot organisms, adaptation, self-adaptation, self-development, long-term artificial evolution.**

I. Introduction

Adaptability and self-adaptability represents an important characteristic of systems working in real environments [1][2]. Different uncertainties, variation of parameters or even an appearance of unknown situations require such mechanisms, which allow the system to find a compromise between achieving the main goal, set by a designer, flexible behavior to fit the environment and self-developmental features, expressed by a self-concept. Finding this compromise requires three important mechanisms: plasticity of the system itself [3]; regulative mechanisms, which uses system's plasticity to perform adaptation [4][5]; and, finally, a goal and self-concept, which drive the system along adaptive and self-developmental changes.

Plasticity of the system can be achieved by exploiting the principle of heterogeneous multicellularity [6]: each module is compatible with other modules and they can assemble and disassemble themselves into structures with different functionality [7], see Figure 1. Not only structural functionality, but also regulative and homeostatic mechanisms can be self-developed; they are addressed by developmental robotics. Multi-robot systems with high-developmental plasticity are explored in several research projects, e.g., [8][9], in common they are referred to as artificial organisms [10].

Structures and functionality of artificial organisms are closely related to each other, by changing macroscopic structure, the system also changes its own functionality and correspondingly behavior [11]. Relation between structures, functions and behavior can be represented as shown in Figure 2. We denote this relationship as "generating" because the upper level generates the lower level, i.e., structures generate functions and functions generate behavior. Controllers in functions and structures – they represent *the regulative level* – allow some degree of flexibility (adaptability) for the system. The targeted behavior on the regulative level is expressed by a "goal", which describes aims of the system itself and the criteria for achieving adaptive behavior. Adaptivity on this level depends on capabilities of a designer to foresee possible environmental changes and to integrate a reaction on these changes into the controllers. To react on unpredictable changes on the design stage, the second *generating level* is required, which can modify controllers. The generating level contains different deriving and evolving mechanisms, which can generate the regulative level and essentially, when not completely, change the system. The targeted behavior on the generating level is expressed by a "self-concept", which is defined in a broader way than a "goal". It describes possible developmental changes and determines what the system "may be". When the behavior on the regulative level is referred to "adaptive", the generating level is associated with "self-adaptive". Such changes on the generating level, which are not directly related to adaptation, but rather to ontogenetic self-modification, can be also associated with "self-development", which originates from the neuroscience community, e.g. [12]. Despite "self-development" is a more general notion, targeting primarily cognitive capabilities, the "self-adaptation" and "self-development" overlap in several points since they are developed in parallel.

Technical systems possess goal-oriented behavior, but should be also adaptive to uncertainties and changes in the environment and have some degrees of freedom for self-development. To some extent, these systems are driven by two different forces: by a goal and by a self-concept. When the degree of adaptation is low, there are no essential conflicts between them. When the plasticity is high, the system can be hindered by self-adaptive processes from reaching the main goal. Here we are facing a new conceptual problem about long-term controllability of self-adaptive and self-developmental processes. Obviously, that either the goal should be formulated in such an invariant way, which allows multiple approaches for its achieving, or self-adaptive processes should basically be limited.

This paper extends and generalizes several ideas expressed in [1][13][14] and introduces a more detailed link between a high-level notion of adaption and practical implementation of adaptive mechanisms. The rest of this work is structured in the following way: firstly, a short introduction into developmental robotics is made in Section II and then different adaptive and self-adaptive mechanisms are overviewed in Sections III and IV. Adaptation and self-adaptation, as well as bound and unbound self-concepts are discussed in Section V. To exemplify the mentioned ideas, the Section VI introduces the problem of macroscopic locomotion for artificial multi-robot organisms. Adaptability of macroscopic locomotion is approached with four main mechanisms: adaptive multi-functional local drivers in Section VI-A, adaptive self-organization on the level of interacting structures in Section VI-B, evolving by using a global fitness evaluation in Section VI-C and a generation by using the bound self-concept in Section VI-D. Finally, this paper is concluded in Section VII.

II. Developmental Robotics

Artificial developmental systems, in particular developmental (epigenetic) robotics [3], are a new emerging field across several research areas – neuroscience, developmental psychology, biological disciplines such as embryogenetics, evolutionary biology or ecology, and engineering sciences such as mechatronics, on-chip-reconfigurable systems or cognitive robotics [15]. The whole research area (not only of artificial systems) is devoted to an ontogenetic development of an organism, i.e., from one cell to a multi-cellular adult system [16].

A closely related field is evolutionary robotics [17], which uses the methodology of evolutionary computation to evolve regulative structures of organisms over the time. Evolutionary robotics tries to mimic biological processes of evolution [18], but also faces challenges of embodiment [19], reality gap [20], adaptation [21] or running on-line and on-board on a smart microcontroller device [5].

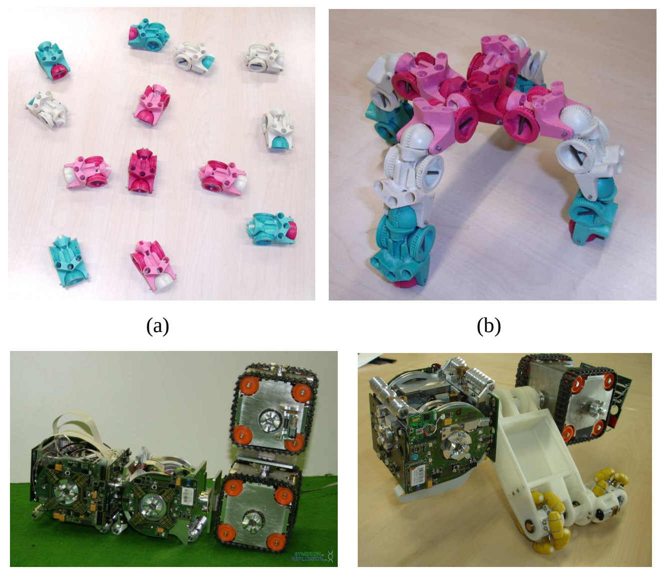
In several points the developmental and evolutionary methodologies differ from each other:

- "*... should try to endow the [developmental] system with an appropriate set of basic mechanisms for the system to develop, learn and behave in a way that appears intelligent to an external observer. As many others before us, we advocate the reliance on the principles of emergent functionality and self-organization ...* " [3];
- "*evolutionary robotics is a new technique for the automatic creation of autonomous robots. Inspired by the Darwinian principle of selective reproduction of the fittest, it views robots as autonomous artificial organisms that develop their own skills in close interaction with the environment and without human intervention*" [17].

Despite differences, evolutionary and developmental approaches share not only common problems, but also some ways to solve them, it seems that both are merging into one large area of self-developmental systems [10].

Both developmental and evolutionary methodologies impose a set of prerequisites on a system; one of the most important from them – it should possess a high degree of *developmental plasticity*. Only then an organism can be developed or evolved. Developmental plasticity requires a specific flexible regulative, homeostatic, functional and structural organization – in this point evolutionary/developmental systems differ from other branches of robotics. Since collective systems, due to their high flexibility and cellular-like organization, can provide such a versatile and re-configurable organization – collective robotics is a suitable object for application of evolving and developmental approaches.

The approach, used in our work, is based on modularity and reconfigurability of the robot platform, as shown in Figure 1. Individual modules possess different functionality



(a) (b) (c) (d)

Figure 1. **(a)** Prove-of-concept: individual robots; (b) Prove-of-concept: aggregated robots into an artificial organism; (c), (d) Real prototypes: aggregated robots (images ©SYMBRION, REPLICATOR projects).

and can dock to each other. Changing the way of how they are connected, an aggregated multi-robot system (organism) possesses many degrees of structural and functional freedom. Due to a capability of self-assembling, robots have a control over their own structure and functionality; in this way they can emerge different "self-*" features, such as self-healing, self-monitoring or self-repairing. These self-* features are related in many aspects to adaptability and evolve-ability, to emergence of behavior and to controllability of long-term developmental processes. The self-issues are investigated in manufacturing processes [22], distributed systems [23], control [24], complex information systems [25] or cognitive sensor networks [26].

Flexibility and changeability of structures and functions are one of the most important aspects of artificial multi-robot organisms [11]. In Figure 2 we demonstrate the dependencies between structures and functions, as well as introduce a two-layers control architecture with regulative and generating levels. The first level is related to control,

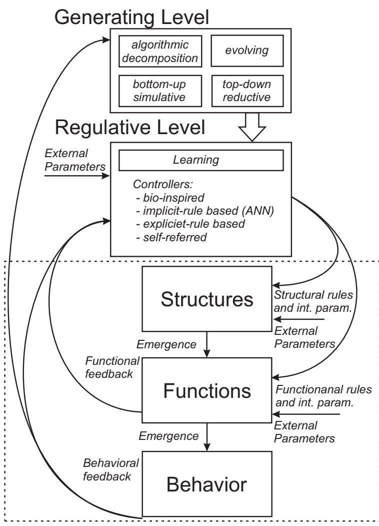


Figure 2. Functional scheme of regulative and generating levels in structural systems (image from [14]).

we denote it as the *regulative level*. It contains different controllers, such as explicit and implicit rule-based (artificial neural networks), different bio-inspired, self-referred or learning systems. These controllers influence structural or functional rules as well as change parameters of the corresponding level. All controllers work based on the scheme: *change of input parameters* $\rightarrow$ *changes of output parameters/rules*. The main goal of the regulative level is to maintain an internal homeostasis of the system, to execute different tasks or, more generally, to demonstrate purposeful behavior depending on external conditions. Controllers at the regulative level allow some degree of adaptability, defined by design goals.

In detail, it depends to which extent a designer of these controllers was able to foresee possible changes of the environment and to integrate a reaction on these changes into the controllers. The controllers allow different degrees of reaction on changes. However, the system at the regulative level is able to react only to changes whose parameter range was predicted in advance during the development of controllers or learning mechanisms. When changes are not predictable at the design stage, we need to introduce the second level, which can modify regulative controllers – we denote this as the *generating level*.

Deriving is primarily related to distributed problem solving and planning approaches, known in the multi-agent community [27], symbolic tasks decomposition [28], structural decomposition [29], self-referred dynamics [30] and others. These approaches are fast, deliver a predictable behavior and can be applied when a new situation is at least structurally known. Evolving is basically related to evolutionary approaches, see e.g. [31], and can be applied when the situation is completely unknown and a large search space of possible solutions should be explored. Recently, evolutionary approaches have been applied to a wide class of robotic problems [32].

Collective systems with such a two-layered control architecture and self-assembling capabilities possess extended developmental plasticity and allow a wide range of adaptation and self-modifications. However, there are several open questions about "driving forces" of adaptation and self-adaptation. Since currently there are several theories towards adaptive systems, we need first to identify classes and mechanisms of adaptation, as it is shown in the next section.

III. Three Mainstreams in Adaptive Systems

Adaptability is often considered in biological terms of natural evolution [33] or environmental uncertainty [34] as well as in management and business processes [35]. There have been several attempts to create a common theory of adaptability, such as the approach suggested by Michael Conrad [34]. Overviewing the vast literature on the field of adaptation, we can recognize three main streams driving further development and representing different methodologies and different approaches to adaptation. The first and oldest stream is related to the theory of adaptive control. Several early works in adaptive control date from the late 50s - early 60s [36][37]. In the mid-late 70s several issues related to temporary stabilities [38] appeared, which in turn led to iterative control re-design and identification, and contributed in the mid-80s to robust adaptive control [39][40]. Overviews of adaptive architectures can be found in textbooks [41][42], which can be generalized as a high-level architecture, shown in Figure 3(a) [43].

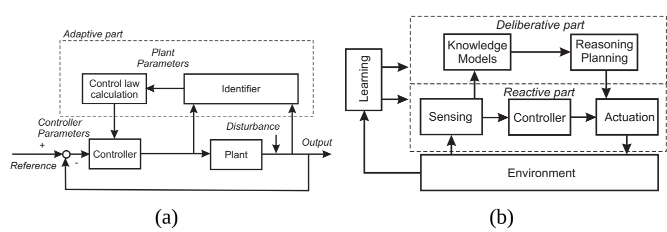


Figure 3. A high-level architecture for **(a)** adaptive control **(b)** adaptive behavioral systems (images from [14]).

Adaptive control consists of two parts, a conventional feedback-based control loop and an adaptive part, depicted by the dashed line in Figure 3(a). The environment is not explicitly integrated into this model, it is implicitly reflected by introducing disturbances and by uncertainties in the plant. The goal of the adaptive part is to estimate the behavior of a plant (by the identifier) and to calculate dynamically the control law (by the control law calculator). When in optimal control, a control law is designed off-line by a designer, an adaptive controller does it on-line. Most challenges in adaptive control theory are concentrated around adaptation of control to parameters of a plant when these parameters are unknown or changing.

The second mainstream of adaptation is located around adaptive behavior, which first arises within the AI community, e.g. [44], and involves cognitive aspects of adaption [45]. There appear a few new components in the scheme from Figure 3(a): explicit environment, sensing and actuation, as well as the deliberative cycle, shown in Figure 3(b). When the reactive part of this scheme is in fact the optimal controller from Figure 3(a), the deliberative part represents a new AI component. The adaptive system is now embedded into the unpredictable/dynamically changing environment; these systems are often referred to as situated systems [46]. Sensing and actuation represent a "body" of the system, intelligence (and so adaptation) is treated in terms of embodiment [47]. Achieving adaptivity in this context is spread into several approaches: different learning techniques in reactive and deliberative parts [48][49][50], behavior-based approaches [51], adaptive planning and reasoning [52], biological inspiration in cognition [53], evolutionary approaches [54] and many others. The goal of adaption can be formulated as achieving desired environmental responses according to some selected fitness/reward criteria.

The third mainstream towards adaptation is related to the community around distributed and software-intensive systems, computational, communication and sensor networks. With some degree of generalization, the business applications can be also related to this mainstream [55]. The environment involves explicit users; the system itself is separated into different levels (applications), which run in parallel [56]. The goal of adaptation here is related to scalability, self-optimization and self-protection, recognition of context, as well as to the software-engineering issues addressing reliability [57].

IV. Mechanisms of Adaptation in Collective Systems

Three mainstreams in the theory of adaptive systems, considered in the previous section, allow making several conclusions towards their underlying adaptive mechanisms. These mechanisms are closely related to three following issues: *developmental plasticity*, *capabilities to determine desired modifications* and, finally, *mechanisms, allowing reaction on changes by utilizing plasticity*. Since adaptive systems are approached from several independent directions (see the first bio-inspired work on adaptation by Ashby [58]), understanding of these underlying mechanisms differs from community to community.

Generalizing experience from the adaptive control theory [42], AI domains [44] and the latest developments in bio-inspired [32] and software intensive systems [59], there are four classes of developmental plasticity:

Plasticity level 1. Fixed interactions. For several applications, mostly in industrial environment, collective agents are expected to work in well-defined environment, where all possible environmental fluctuations can be absorbed by external mechanisms (e.g. by human personal). In this way, it is much cheaper to make agents with fully or partially predefined behavior. Cooperative behavior of collectively working robots includes some number of adaptive mechanisms, however is mostly preprogrammed [60].

Plasticity level 2. Tunable and reconfigurable cases. These collective systems have several degrees of freedom related to developmental plasticity. Adaptivity here is achieved in different ways: from parameter changing, feedback-based mechanisms [61], adaptive self-organization [62] until fully reconfigurable systems. Here also a multitude of learning mechanisms can be applied [63].

Plasticity level 3. Bounded development. Adaptivity is designed to be in some range of possible variations. Normally, it is defined by some structural mechanisms, for example by a nature of reward. The limit of adaptive systems is reached when a new structural change happens or the system is not able to identify the required reward (for reward-based mechanisms). In this case the system needs to modify its own structure to absorb environmental changes. We refer the systems, capable of structural changes with flexible reward/feedback mechanisms, to developmental collective systems (see more in Section V).

Plasticity level 4. Unbounded development. "Unbounded" means a very high degree of developmental plasticity, similar to biological cellular systems. Such systems are potentially capable of unbound increasing of their complexity, diversity or information capacity (see more in Section V).

Considering capabilities to detect changes and to determine desired modifications, we basically refer to three following schemes:

1. Model-reference based detection. This is a widely used scheme in e.g. adaptive control [64], machine learning [63], artificial evolutionary systems [17] and many other areas, where the detection of changes represent an error between a model and a system ("plant" in control theory). Multiplicity of *Feedback-*, *Reward-*, *and Fitness- based* mechanisms [61][65] originate from this model-reference based approach. This is the main detection mechanism for adaptive systems.

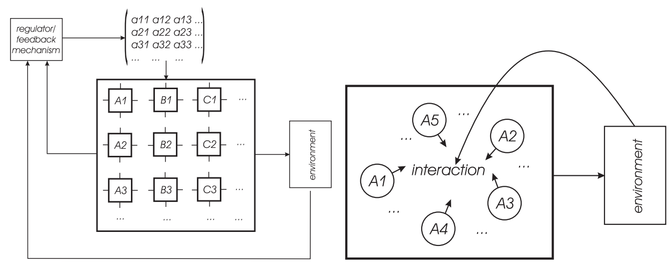
2. Self-tuning based detection. This is also very popular approach, see e.g.[66], the first ideas are referred to [67]. It consists of a parameter estimator, a design calculation and a regulator with adjustable parameters, the idea is to select "a design for known plant parameter and to apply it to unknown plant parameter, using recursively estimated values of these parameters" [64, p.189]. Self-tuning mechanisms are often used in terms of self-adaptive control, especially in the 60x and 70x [68].

3. Concept-based detection. Self-developmental systems with a high degree of plasticity cannot use model- or tuning-based detection mechanisms – mechanisms of detection are not plastic enough to follow these systems. Instead so-called self-concept-based approach has been proposed (first in human psychology [69][70], see more in Section V). This mechanism determines desired modifications based on internal stimuli, containing in the self-concept; in many cases each self-modification creates a new generation of adaptive changes, which are absorbed by the model-reference based detection. This mechanism is mostly utilized in self-adaptive systems.

The mentioned developmental degrees of freedom, together with the detection mechanisms, can create different combinations, which result in several adaptive and self-adaptive mechanisms. Focusing now on adaptation, we indicate three main classes of such mechanisms:

1. Parameter-based adaptive mechanisms. This kind of adaptive mechanisms has a long tradition in the control theory, see e.g. [41]. Here the system is controlled through control parameters, see Figure 3(a). By modifying the values, the controlled system responses by changing its behavior (in the terminology of control theory – the transfer function). There exists a multitude of possible variations: when the system is known, its analytical model can be used for control purposes; when the environment is simple – it is incorporated into the analytical model; when the system is unknown (the black box approach) – different feedback mechanisms can be utilized for control purposes. Different ways of how to adapt the system are the focus of unsupervised reward-based learning approaches. The parameter-based adaptive mechanisms are very efficient, however possess several drawbacks. First of all, the system is adaptive only within such variations of a transfer function, which are allowed by changes of control parameters. The second point is related to the feedback mechanisms/analytical regulator – this element represents a general bottleneck. For example, a feedback mechanism expects only a temperature as a feedback parameter. In a situation where not only a temperature but also a light becomes a feedback parameter, the predefined regulator will not provide an adequate regulation.

2. Modularity-based adaptive mechanisms. To increase flexibility of the systems to react to environmental changes, another principle has been suggested. This principle is based on the so-called "atomic structure", where the system consists of modules, which can be dynamically linked to each other. The linkage can be of binary as well as fuzzy character, see Figure 4. Examples of such systems are artificial neuronal networks (ANN) [71], Genetic Programming (GP) [31], reconfigurable robotics [72] and others. Modular structure has several dedicated issues, i.e., granularity of modules – how large are changes of the transfer function by re-linking only one elementary module. Developers are trying to design the modules so that to make this change as small as only possible – i.e., to provide possibly smaller granularity. Not only the system itself, but also the regulator can be based on a modular structure; this eliminates drawbacks of parameter-based mechanisms related to a fixed structure of regulator.



(a) (b)

Figure 4. Simplified structure of **(a)** modularity-based adaptive mechanisms ; **(b)** self-organized adaptive mechanisms.

3. Self-organized adaptive mechanisms. In contrast to the two previous adaptive mechanisms, the self-organization represents another approach for adaptation. Self-organizing systems consist of many interacting elements with a high degree of autonomy [73], see Figure 4. The transfer function of such systems is "generated" dynamically through interactions. Usually, when these interactions are not synchronized, the transfer function is irregular or even chaotic. However, when these interactions become synchronized in some way, we observe an appearance of "ordered transfer function". Self-organized adaptive mechanisms introduce a feedback directly into the interactions among elements. In this way, any changes in a local feedback modify the whole collective behavior [74][75]. In many cases it happens without any regulators at all; all interacting elements modify their own interactions [76].

Since environmental changes require an adaptive reaction from a system, which in turn requires specific control mechanisms, we can divide changes and reactions into those forecast in advance and correspondingly those not forecast in advance. This division is relative, because in practical situations each change has forecasted and not forecasted components.

Now, based on the introduced concepts, we can define adaptability [14]. **Adaptability** is closely related to environmental changes and the ability of a system to react to these changes and the capability of the designer to forecast reaction of the environment to the system's response. Therefore adaptability is defined in term of the triple-relation: *environmental changes* $\rightarrow$ *system's response* $\rightarrow$ *environmental reaction*. In general, adaptability is the ability of a collective system to achieve desired environmental reactions in accordance with a priori defined criteria by changing its own structure, functionality or behavior initiated by changed

Table I
FOUR TYPES OF ENVIRONMENTAL CHANGES IN ROBOTIC APPLICATIONS AND EXAMPLES OF CASES BOTH FORECAST AND NOT FORECAST IN ADVANCE, FROM FROM [14].

Environmental changes leading to:	Examples: Forecast in Advance	Examples: Not Forecast in Advance
Appearance of new situations	Installation of industrial robots in a new workshop	Work in previously unexplored environment (e.g. landing on Mars)
Changed functionality	Changing a type of locomotion (e.g. from wheeled to legged), when changing a terrain type	Search and rescue scenario when robots encounter unknown obstacles
Modified behavioral response	Gravitational perturbance of flying object in space and finding new control laws for engines	Distributed control of legged locomotion for obstacles of random geometry
Optimization of parameters	Changing of day/night light and adapting intensity of additional light	Adapting locomotive parameters for randomly moving obstacles

environment. In Table I we roughly specify four different categories of environmental changes.

According to environmental changes from this table, we can identify five different classes of adaptability in collective systems, capable of structural phenomena: optimization mechanisms; behavioral control; functional control; derivation of new regulatory functionality and, finally, evolving of new regulatory functionality. These mechanisms are graphically represented in Figure 5. Since we involve in this figure

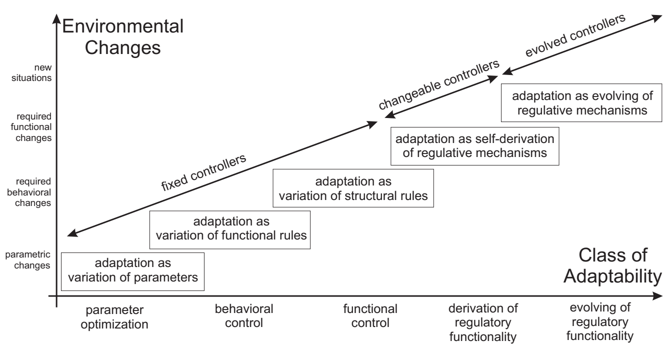


Figure 5. Different adaptivity mechanisms in collective systems, from [14].

several evolutionary mechanism, we closely touch the issues of self-adaptivity, considered in the next section.

V. Adaptation vs. Self-adaptation

The ideas, expressed in the previous sections are related to adaptation. There are several differences between adaptation and self-adaption. For example, Bäck in [77] distinguishes between *dynamic parameter control*, *adaptive parameter control* and *self-adaptive parameter control*. Here "self-adaptive" includes (evolutionary) mechanisms for changing regulative structures, whereas "adaptive" means merely feedback-based regulative mechanisms. This and similar definitions of self-adaptivity is widely used in evolutionary [78] and in autonomic [79] communities.

The theory of adaptive control also uses the term of self-adaptation, however in another context. It is primarily related to different variations of well-known self-tuning mechanisms [67][66], where the detector and regulator uses iterative approach for identification of control laws. On the early stage of 60x and 70x the term of self-adaptation was widely used, e.g. [68], whereas modern literature refers self-tuning approaches to adaptive systems [64].

Taking into account regulative and generating levels from the Figure 2, the difference between adaptive and self-adaptive seems to be more complex. We start from the commonly accepted fact, that "adaptive" and "self-adaptive" are placed on different levels of hierarchy. In Figure 6 we draw these two levels (as two dashed boxes).

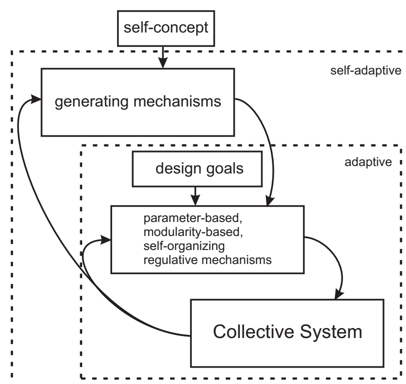


Figure 6. Adaptive and self-adaptive mechanisms.

The mechanisms in the first box allow adaptive behavior, related to the design goals of a system. These "goals" are implicitly formulated as a transfer function, model-reference and other mechanisms related to behavior in environment. In the same manner, the self-adaptation needs also a "goal", however the self-adaptive goal should be expressed in a more broad and flexible way and should be related to the system itself. It describes developmental goals as "what a system may be", instead of "what a system should be". To explain the difference between both, we can consider the case of macroscopic locomotion for such an organism, as shown in Figure 1(b).

This organism can have a series of specifications: legged or wheeling principles of motion, specific limitations imposed on energy consumption or on a structural stability. They can be formulated even broader, as e.g. "a system capable to move from A to B", and expressed as a fitness function of the traveled distance and constrained by e.g. segmented (such as insects) construction of body, symmetric movement of legs or humanoid-like structure of body. All of them are different examples of design goals, which, when touching with reality, produce some adaptive locomotive structures and behavior. Obviously, that formulating design goals more or less broad, we allow more or less degrees of adaptability.

Now, we can assume, that driven by human developmental history, we have a specific vision of how this organism may be: complexity of regulative and homeostatic functionality, degree of "intelligence", flexibility of structural reconfiguration, scalability. They can be also broader, such as "increasing of information capacity". These visions are not directly related to locomotion, they express some desire of how to see the whole systems. From these visions, it is not always possible to obtain some locomotive structures directly. Each change along the "vision axis", requires follow-up changes along "adaptive axis". In other words, "self-vision axis" and "adaptation axis" are different. Such a "vision of itself" is expressed in terms of a "self-concept". The notion of self-concept originated in human psychological research, e.g. [70], and is basically related to human self-developmental processes, e.g. [69][70][80]. Recently, there appear several works, which apply psychological ideas to robotics, e.g. [81][82].

In self-adaptation we have to point out one principal element, related to the bounded and unbounded character of changes. When in adaptive processes, these driving forces are mostly bounded, expressed e.g. by reward or fitness, the self-concept may include driving forces, which are of unbounded character. **Self-adaptation** can be formulated as a series of changes, undertaken by the system alone, and intended to adapt to its own vision of itself.

When the self-concept has an unbound character, the system is in fact continuously changing itself. In this context, sell-adaptation can be related to another process – self-development. The notion of self-development in robotics is most probably originated from another community – neuroscience, e.g. [12][83], which through artificial neural networks (ANN), e.g. [84] and evolutionary communities find its own way to robotics, e.g. [82]. The development focuses on ontogenetic processes related to cognitive science and the concept of embodiment [3], whereas self-development is understood more broadly as e.g. self-exploration, self-supervision, self-learning and others. To be consistent with the logic of these notions, the **self-development** is a more general ontogenetic mechanism of continues changes, which targets cognitive aspects and may be unlimited in time and complexity, i.e., it possesses unbounded properties. Both self-adaptation and self-development are related to the self-concept.

In evolutionary community unbounded properties are often related to open-ended evolution, which *is characterized by a continued ability to invent new properties – so far only the evolution of life on Earth (data partly from the fossil record) and human technology (data from patents) have been shown to generate adaptive novelty in an open-ended manner* [85]. We find some first ideas about open-ended evolution in [86] and [87]. Open-ended evolution is also related to indefinite growth of complexity [88] and unbounded diversity [89]. Ruiz-Mirazo and co-authors expressed the interesting idea that *"the combination of both self-assembly and self-organization processes within the same dynamic phenomenon can give rise to systems with increasing levels of molecular as well as organizational complexity"*. They also proposed to decouple genotype and phenotype from each other. A similar idea of increase homeostatic autonomy in macroevolution was proposed by [90], which leads us to not-fitness driven self-developmental processes. Several implementations of open-ended evolutionary scenarios, e.g. [91], do not use any explicit behavioral fitness, moreover, there is no complexity growth in such "classical" artificial life simulator as Tierra and Avida [92]. In this work Russell Standish proposed to improve these systems: "*a key step in doing this is to generate a process that adaptively recognises complexity, since it will be impossible to include humans in the loop, even when run on conventional computing platforms*".

These works lead us to an interesting question about the unbounded self-concept: which process can generate complexity? One of the first remarks is from von Neumann: "*synthesis of automata can proceed in such a manner that each automaton will produce other automata, which are more complex and of higher potentialities than itself*" [86]. A similar approach is observed in L-Systems [93] (authors used evolutionary process but human operator in the selective loop) as well as in self-referred dynamics [30]. It seems that structural production can lead to growth of complexity and diversity.

However, considering the Kolmogorov complexity of fractal structures, which is equal to the shortest production set of rules [94], we note the complexity of the whole fractal is independent of its size – the self-similar structural production does not increase complexity. Thus, we require that production systems include parameters, which perturb generating structures. In this way, structural production rules parameterized by a random (environmental) value may lead to infinite growth of complexity and diversity, and are candidates for the unbounded self-concept. In Table II we collected several possible self-developmental processes in structural collective systems with bounded and unbounded self-concepts.

To conclude this section, we argue that adaptation and self-adaptation are two different, hierarchically placed processes, *related to an origin of changes and not to the used mechanisms (both processes can use the same mechanisms)*. Related to the utilized degrees of plasticity and origin of modification, different adaptive and self-adaptive mechanisms can be combined into three groups and represented as shown in Figure 7.

Design goals and self-concept also differs from each

Table II
SEVERAL CHARACTERISTICS OF SELF-DEVELOPMENTAL PROCESSES IN COLLECTIVE SYSTEMS, FROM [10].

Process	Developmental plasticity	Self-Concept
Regulatory	Structural and functional plasticity of the system, controllers can change their own transfer functions.	*(bound)* Achieving a targeted goal in changing environment. *(unbound)* Increasing performance characteristics.
Homeostatic	Like in the regulative case, but related to maintaining steady internal states in changing environment.	*(bounded)* Endogenous steady state. *(unbounded)* Achieving best possible homeostasis for diverse scalability metrics.
Learning	Changeable structure of regulative system.	*(bounded)* e.g. positive or negative rewards. *(unbounded)* Fitting very large (infinite) parameter space, e.g. by exploring structural-functional relations.
Planning-driven	Structural, functional and regulative plasticity.	*(bounded)* Minimizing deviations from a plan. *(unbounded)* Self-referred planning.
Fitness-driven	Structural, functional and regulative plasticity.	*(bounded)* Explicit fitness. *(unbounded)* Implicit fitness (optimizing energy balance, maximizing off-springs).
Open-ended	Capability for unbounded evolutionary activity.	*(unbounded)* Unbounded metrics.

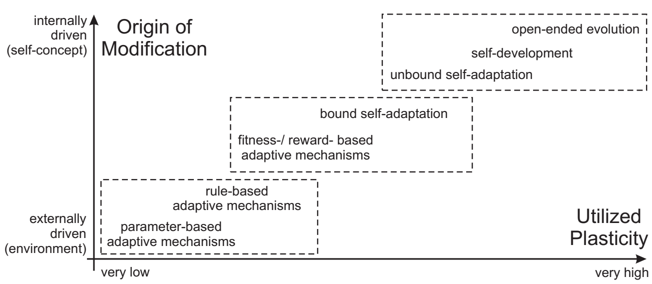


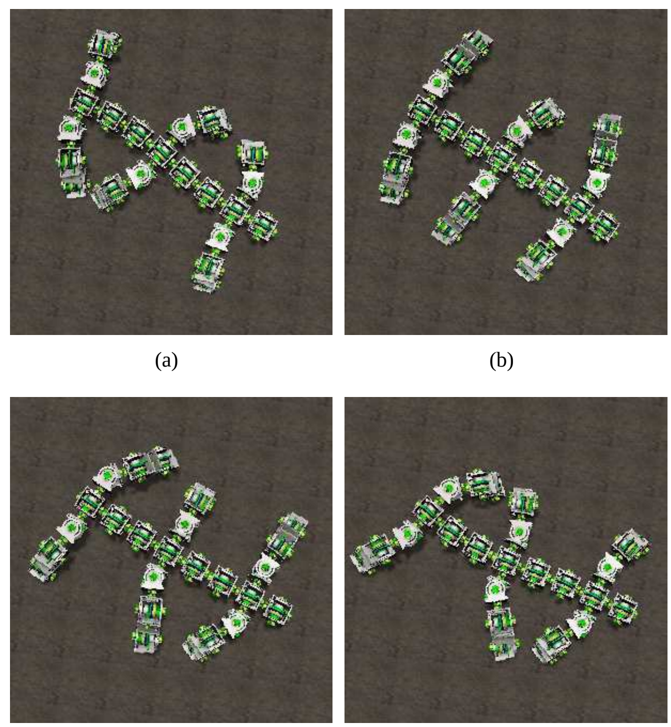
Figure 7. Three groups of adaptive and self-adaptive mechanisms placed along the used degrees of plasticity and origin of modification.

other; self-concept is more "system common" description and has more degrees of freedom. Normally, during adaptation, a system cannot change its own goal. However, during self-adaptation, a system can potentially change the design goals, i.e., self-adaption and goals can potentially be conflicting. When the plasticity is high, and the system can be hindered by adaptive processes from reaching the main goal, we are facing a new conceptual problem of a long-term controllability of adaptive and self-developmental processes.

There are several strategies to avoid conflicts between achieving design goals and self-adaptation. One of them is to formulate the self-concept invariant to possible adaptations. There are several mechanisms expressing such an invariant property of the generating level: symmetries, conservation laws or e.g. "templates". Templates are well-known in cognitive science [95] (also as "schemas" or "prototypes"), in topological research (in knot and braid theory) [96], as well as known as "frames" in the AI community [97]. The self-concept can be also expressed by symmetries, conservation laws, be planning- or fitness-driven or even have a character of unbounded metrics for open-ended evolution.

VI. APPROACHING ADAPTABILITY OF ARTIFICIAL ORGANISMS FOR COLLECTIVE LOCOMOTION

To exemplify the discussed concepts of adaptation, self-adaptation and self-development, we consider the problem of collective locomotion for multi-robot organisms. Figure 8 shows a top-down view on a hexapod organism. The whole



(a) (b) (c) (d)

Figure 8. Top-down view on a hexapod multi-robot organism from Figure 9(a). Shown are four different positions of legs to illustrate a complexity of collective locomotion.

organism represents a collection of aggregated modules, which form the central vertebral column and six legs connected to the spine. Since 1DoF modules are connected in vertical and horizontal planes, legs as well as the vertebral column possess multiple angular and dispositional degrees of freedom, required for the legged locomotion. This organism moves on a flat surface without any obstacles; four different frames of this movement are shown.

From these images it is well visible, that the regular motion patterns (when there are no obstacles) can be split onto three different parts:

(a) Periodical (or rhythmic) activation of "active joints" (elements, which connect legs to the vertebral column).

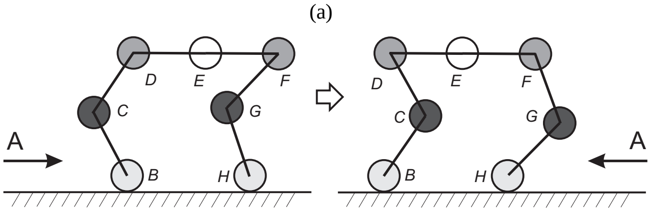
(b) Motion of legs in a vertical plane. This patterns is basically the same in all legs.

(c) "Rippling" of the vertebral column. The amplitude and frequency of this motion should be synchronized with active joints.

Basically, the motion without obstacles represents a classical controlling problem, which can be solved with e.g. kinematic analysis [98], evolved [32] or resolved by using bio-inspired approaches [99]. The problem of adaptation appears first, when an organism should overpass some obstacle, and this requires multiple co-depending changes of patterns (a)-(b). Moreover, the works on CPG, e.g. [100], indicate that any adaptive modification of the macroscopic multi-cellular behavior requires multiple correlations between individual degrees of freedom and, in the worst case, may essentially increase the complexity.

The need of multiple synchronization may be better understood in Figure 9, which shows the 2D section of an aggregated organism with several active joints in the front (two front legs). We assume that this structure is already created

(a)



(b)

Figure 9. **(a)** Artificial organism, front view; **(b)** 2D section of an aggregated organisms (two front legs with a section from vertebral column), circles are active joints. Shown are two kinematic states of this organism.

(or evolved) and represents some optimum of functionality for a locomotion without obstacles. Each of the aggregated modules possesses independent motors (degree of freedom, displayed by a circle) and can actuate independently of each other. In order to move as an organism, all these motors should perform synchronized individual actuations. There are several requirements, such as:

- the center of gravity should not overstep the nodes B and H, other case the organism will be unstable;

- even in homogeneous case there are several non-symmetries caused by differences in docking elements, or more generally by different modules. This leads to non-symmetrical positions of several active nodes, like C and G;

- we require that some structural nodes are e.g. strongly horizontal (vertical) as e.g. D, E and F.

- all nodes have different load. This is indicated by different gray level of active nodes.

Each motor is controlled by a non-linear rhythmic driver, whose control parameters depend on internal sensors (e.g. torque of a motor). Without loss of generality, we say this represents a simple adaptive control on the functional level, where motors are first not connected with each other. This scheme is sketched in Figure 10. Now, we insert a

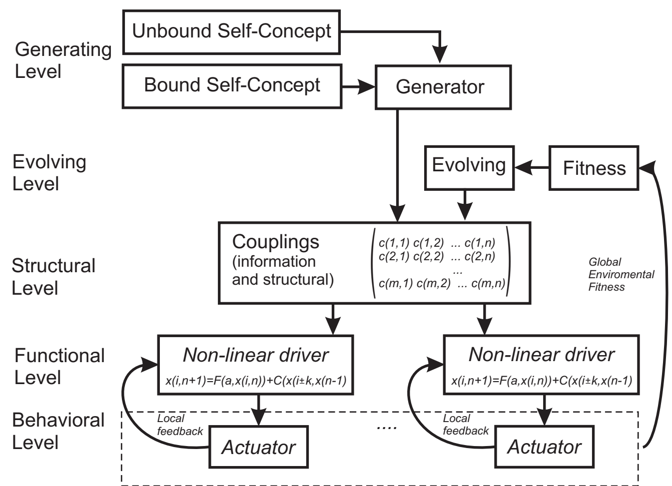


Figure 10. Different levels of adaptive collective locomotion.

structural level, which depends on a morphology of the organism. This level is represented by a coupling element $\underline{\underline{\mathbf{C}}}$, which creates "communication channels" between different nonlinear drivers (there are several different coupling elements $\underline{\underline{\mathbf{C}}}$ on e.g. structural and information levels). Since organisms create generally three dimensional structures, we expect at least a coupling between three elements (as e.g. a tensor of the third order). The coupling element contains values like $c_{ijk} = 1$ (direct coupling between drivers i, j, k), $c_{ijk} = 0$ (no coupling between drivers i, j, k), $c_{ijk} = -1$ (phase inversion between drivers i, j, k) or even any positive (amplification) or negative (decay) coefficients. Collective actuation depends on coefficients in these coupling elements.

As mentioned above, any non-periodical perturbation, e.g. motion with obstacles, requires multiple synchronization between elements, which firstly adapt the collective actuation of all motors; secondly takes into account stability constraints. There are three different mechanisms, which can be used in creating adaptive structure and functionality around $\underline{\underline{\mathbf{C}}}$. Firstly, individual rhythmic drivers use local adaptive mechanisms, know in the theory of adaptive control, as shown in Section VI-A. Secondly, drivers and $\underline{\underline{\mathbf{C}}}$ represent a coupled map lattice (CML) [101]. As we can see from Figure 9, nodes B, C, D, F, G, H have the most intensive load, which can lead to a more stronger synchronization in $\underline{\underline{\mathbf{C}}}$, where as other nodes do not need any synchronization and their connection will disappear. In this way, synchronization effects in CML represent an emerging adaptability created by self-organizing processes between behavioral, functional and structural levels. This effect is similar to the observation in CPG with environmental coupling [102]. This approach is sketched in Section VI-B. Then, a structure of $\underline{\underline{\mathbf{C}}}$ (and so a collective locomotion) can be evolved, as described in Section VI-C. Here we face the problem of deriving such local and global fitness functions, which adapt a collective actuation within the framework of constrains.

The processes, mentioned above, lead to an adaptive macroscopic locomotion, e.g. when an organism encounters an obstacle. However, changes in collective actuation can be occurred even when an organism does not encounter an obstacle (just to remind this organism already reaches some optimum in fitness, i.e., these changes cannot be driven by an "old fitness"). To initiate such changes, we have to introduce a new "driving force", which is independent of particular obstacles. This will be then a self-adaptation, which takes place on the generating level, as shown in Figure 10. There are several proposals for bound and unbound self-concepts as shown in Section VI-D.

A. Multi-functional, Locally Adaptive Rhythmic Motor Driver

As described in the previous section, individual motor drivers should demonstrate diverse dynamic behavior. In literature there are known different types of continuous rhythmic generators, e.g. [102], however due to technological reasons of controlling DC motors and running on a small microcontroller, we prefer time-discrete systems. Each time-step can be selected as small as possible, for example a few μsec to guarantee a quality of control. Dynamic variables, e.g. x_n, represent voltage (current, phase), which are applied directly to DC-DC convertor or H-bridges. To obtain diverse dynamics, we use the idea of changing the determinancy order of normal form (NF) and the following perturbation of nonlinear terms [103]. This can be achieved when to use hierarchical non-homogeneous coupling for any well-know low-dimensional system, for example the logistic map. This approach is very common in the community (e.g. [104][105]). In our case, the map has the following form:

$$\begin{aligned} x_{n+1} &= cy_n + ax_n(1-x_n), \\ y_{n+1} &= cx_n + bx_ny_n(1-y_n), \end{aligned} \tag{1}$$

where $x_n \in \mathbb{R}$, $y_n \in \mathbb{R}$, c is the coefficient of the linear coupling, b is the coefficient of the nonlinear coupling, a is the general bifurcation parameter. As turned out, the dynamics of (1) in fact has little in common with the initial logistic maps. The system (1) is denoted as the ordinary logistic-logistic (OLL) map. Several examples of qualitatively different types of behavior are shown in Figure 11.

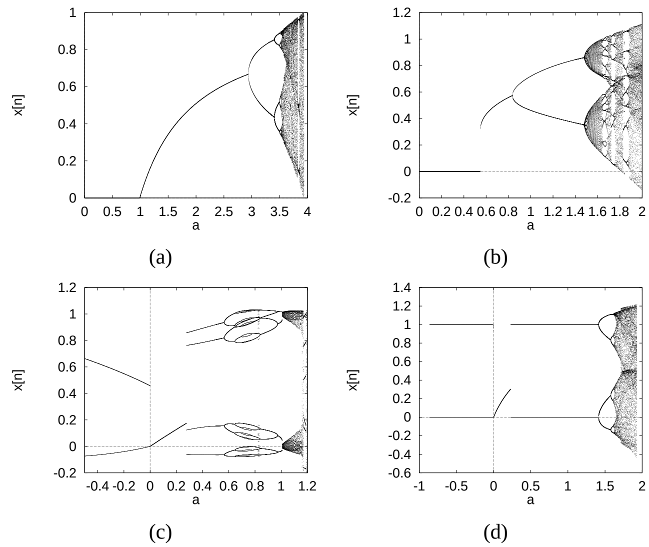


Figure 11. Several examples of qualitatively different types of behavior of the system (1). Bifurcation diagrams of the OLL map (1) at parameters: **(a)** $b = 1$, $c = 0.1$, $x_0 = 0.1$; **(b)** $b = 2$, $c = 0.6$, $x_0 = 0.4$; **(c)** $b = -1.5$, $c = -1$, $x_0 = 0.1$; **(d)** $b = -0.5$, $c = -1$, $x_0 = 0.1$.

As shown in [106], the non-homogeneous coupling in (1) increases determinancy order of initial NF. This can be understood as a perturbation of the original logistic map by couplings. In order to obtain all possible perturbed nonlinear terms, it needs to calculate the universal unfolding that is given e.g. by

$$G(\varphi_n, \lambda_u) = \alpha_1 + \lambda_u\varphi_n + \alpha_2\varphi_n^2 + \alpha_3\varphi_n^3 + \alpha_4\varphi_n^4 + \varphi_n^5 \tag{2}$$

with the codimension 4, where α_i are coefficients. We can see that non-homogeneous coupling method of OLL map changes the codimension of local bifurcation from 1 (transcritical bifurcation contained in the logistic map) to 4.

The approach (2) can be used for designing a programmable series of bifurcations so that to create a desired dynamics of the system (1). This allows us to use this system directly in the mechanisms of local adaptation. Coefficients a, b and c can be connected to locomotive sensors (for example a torque sensor). When a load on motor is increased, a local control mechanism (e.g. PID regulator [41]) adapts the coefficient a, e.g. to achieve the required torque on the given load. In the next section we will see several adaptive effects, which arise when many of individual motor drives, like (1), are connected into one system.

B. Adaptive Mechanisms Based on Self-organization

Considering modular robots with the ability to dock to each other and to build multi-robot organisms, the problem occurs how to synchronize the behavior and especially the collective locomotion for different organism's topologies. Traditionally, such problems have been treated by using classical model-based methods. The developed controllers either use such model-based approaches or utilize bio-inspired or evolutionary algorithms. However, most of these approaches are not fully applicable for a large scale modular robotics because of a very high complexity, the huge amount of exchanged data and limited hardware capabilities. Most algorithms fail also due to the lack of scalability and adaptability. The development of new techniques for adaptive treatment of such problems is required.

In the last decades, several approaches from the field of non-linear dynamics have been applied to robotics, especially to solve the problems of locomotion in bipedal [107] or multi-legged robots [108]. The big challenge is still the synchronization between joints or legs so that the generated locomotive pattern become adaptive to environmental changes. Stable attractors provide often the best way to develop a system, which is able to generate several patterns by low-dimensional coupled equations with only a few control parameters. In multi-body systems with many degrees of freedom such methods allow reflecting the real dynamics only in a very limited way. Several attempts have been undertaken to use feedbacks in time-delayed nonlinear oscillators [109][110] or feedbacks based on resonance effects [111][112]. Such feedbacks can address several local and global properties of the dynamics, however currently achieved results target often very specific problems and lack in generalization to other applications. In this section we present an approach based on the Coupled Map Lattices (CML) [101], which focus on synchronization effects achieved in high-dimensional coupled equations.

Each site in the CML is considered as a unit (joint angle, hinge motor, link), which can be coupled with their neighbors through a coupling parameter. We use three different coupling structures: unidirectional or bidirectional coupled rings and four-connections-sites on a 2D lattice. Synchronization between the robots appears through the synchronization effect of spatiotemporal chaotic pattern, modeled by oscillating nonlinear equations. When synchronization appears in a region of the CML, this means that the communication between robots in this region is rapidly decreased and the corresponding part of the organism performs a synchronized movement.

In the first scenario we analyzed the one-dimensional ring-coupled topology (Figure 12). As a "basic" system we use the homogeneously coupled logistic map (Figure 11 (a)). Each site can be additively coupled with their left and/or right neighbor sites either in unidirectional or in bidirectional way. Synchronization appears due to interactions between non-identical systems, which leads to a locking of their phases, whereas their amplitudes remain uncorrelated. As the first test system we take the unidirectional ring map lattices of the length m:

$$\begin{aligned} x_{n+1}^i &= (1-\varepsilon)f(x_n^i) + \varepsilon(f(x_n^{i-1})) \\ x_n^{i+m} &= x_n^i \end{aligned} \quad (3)$$

where $x_n \in \mathbb{R}$, $i = 1, 2, ..., m$ and n represent the dimensions of the CML. $f(x_n^i)$ is the logistic map

$$f(x_n^i) = a x_n^i (1 - x_n^i). \quad (4)$$

Important parameters are the small coupling parameter ε that denotes the strength of nearest neighbor coupling and the bifurcation parameter α.

Figure 12. Multi-robot organism connected to a ring.

Experimental results show that the synchronization between sites occurs within $0.16 \leq \varepsilon \leq 0.19$, observable as bright areas in Figure 13. During the iteration process, in order to simulate a disturbance, we apply a small fluctuation in the bifurcation parameter α, which can be e.g. associated with a disturbance in the communication load. Such a disturbance in turn can be referred to some disturbances in the environment (i.e., obstacles, environmental changes or sensorimotor disturbances). In Figure 13, the bifurcation parameter was slightly disturbed for a short time period and after few time steps when perturbation stopped, the system becomes again synchronized (area in boxes).

Further analysis of the local and global impact has been done by investigating the impact of disturbances in small separated regions (Figure 13 (a)) or if the disturbances appear in local neighborhoods (Figure 13 (b)). As it can be observed in these figures, better synchronization effects occur if a perturbation appears in the sites that are close to each other (local impact). In a multi-robot organism, this means that units in a local range (one leg, arm etc.) perform better synchronization than robots far away from each other.

In the next test scenario we extended the model by coupling the sites with both left and right neighbors (bidirectional coupling). We took again a ring map lattice of the length m

$$\begin{aligned} x_{n+1}^i &= (1-\varepsilon)f(x_n^i) + \tfrac{\varepsilon}{2}(f(x_n^{i-1}) + f(x_n^{i+1})) \\ x_n^{i+m} &= x_n^i. \end{aligned} \quad (5)$$

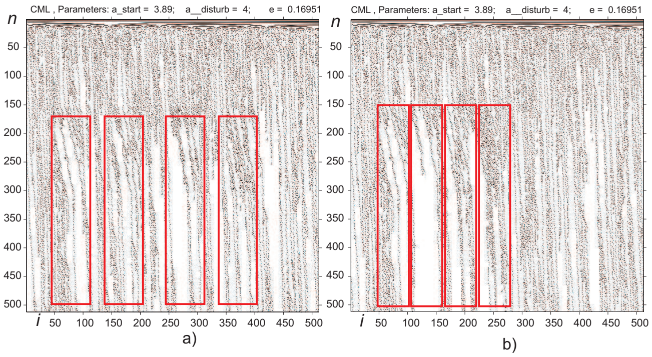


Figure 13. One-way coupled map lattice. Initial conditions: $x_1^i = 0.1 + 0.0000049i$. Boundary conditions: $x_1^i = x_m^i$; **(a)** disturbances (i.e. motor voltage drops) are simulated in a large distance and do not significantly affect each other; **(b)** disturbed sites are in local range and affect more each other than in (a).

In order to get a homogeneous coupling the coupling parameter ε is divided by two. We observe similar synchronization effects like in the previous experiment in sites that are nearby or far away from each other.

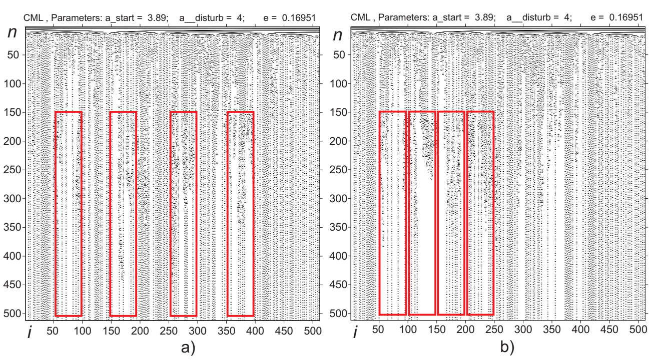


Figure 14. Two-way coupled map. Initial condition: $x_1^i = 0.1 + 0.0000049i$. Boundary condition: $x_1^i = x_m^i$; **(a)** disturbances are simulated in a large distance and do not significantly affect each other; **(b)** disturbed sites are in local range and affect more each other than in (a).

Open or closed chains are the basic structures in many robotic systems since robots often consist of legs and arms. In simulation above we analyzed and developed mechanisms focusing on achieving a synchronized locomotive behavior for multi-robot organisms with only chain-like structures. Now we extend this idea for multi-legged robots with three dimensional topologies. To achieve this, the dimensionality of Equation 5 increased. In this case, the coupling can be performed in all four directions

$$\begin{aligned} x_{n+1}^{i,j} = \quad & (1-\varepsilon)f(x_n^{i,j}) + \tfrac{\varepsilon}{4}(f(x_n^{i-1,j}) \\ & +f(x_n^{i+1,j}) + f(x_n^{i,j-1}) + f(x_n^{i,j+1})) \\ x_n^{i+m,j+m} = \quad & x_n^{i,j}, \end{aligned} \tag{6}$$

where $i, j = 1, 2, ..., m$ represent the system's size. For each time step we generate a separate 2D lattice and perform the time analysis (Figure 15).

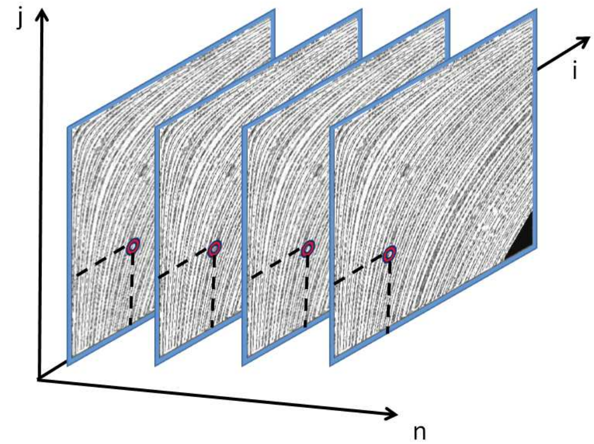


Figure 15. Time evolution through 2D CML sites.

Like in experiments with serial couplings, in order to analyze synchronization properties of 2D spatial lattices, we temporary disturb the bifurcation parameter in a block-shaped regions (Figure 16). It can be considered e.g. as a disturbed part of a multi-robot organism (legs, arms etc.).

The results in Figure 16 show that a small perturbation of α does not cause a chaotic behavior like in the previous experiments for serial coupled sites, but leads instead to a phase synchronization. The reason is the asynchronous updating of the sites [113]. This approach not only synchronizes locomotive behavior but also allow forcing the organism to change the locomotion pattern. In Figure 17 different conceptual layers for the whole framework are introduced: Couplings-, CML-, Actuator-, and the Organism Layer.

In the Couplings Layer (Structural Level), we generate the coupling matrix $\underline{\underline{\mathbf{C}}}$ mentioned in Section VI. This matrix maps the topology of the multi-robot organism by inserting ones and zeros as matrix elements. According to the structure of the coupling matrix the corresponding areas in the CML are activated (one) or not activated (zero). On CML Layer (Functional Layer), we perturb the sites in the activated areas from the Couplings Layer and can observe phase propagation and as well as active phase shifting effects if required (Figure 16). All non-activated sites in the CML can be considered as virtual modules, which do not exist in reality, however are required in order to generate the phase synchronization patterns.

At the Actuator Layer (Behavioral Layer), standard controllers (PI, PID control etc.) can be applied and are often a part of servo motors. Such controllers are well-known from the theory of control and enable motors to follow the generated phase patterns. By learning the phase patterns generated by different perturbations in α, we are able to use this knowledge and actively apply it to generate desired locomotion patterns. The bifurcation parameter α is hence

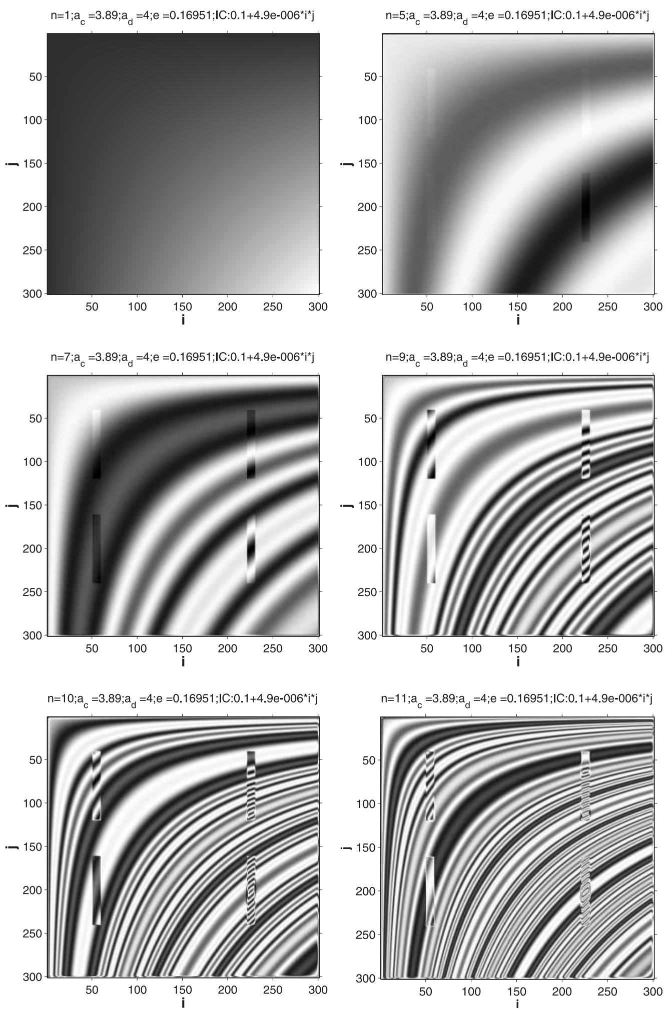


Figure 16. Phase pictures of 2D CML with four disturbed regions.

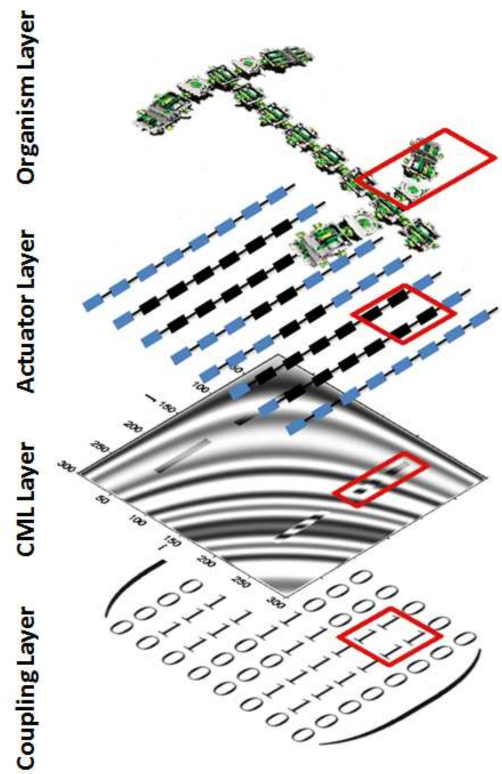


Figure 17. Architecture of Active Phase Perturbation Approach for Locomotion. This figure demonstrates how the CML based approach can directly referred to real robotic applications.

$$
\begin{aligned}
x_{n+1}^{i,j} &= (1-\varepsilon)f(x_n^{i,j}) + \frac{\varepsilon}{4}(f(x_n^{i-1,j}) + f(x_n^{i+1,j}) + \\
&\quad f(x_n^{i,j-1}) + f(x_n^{i,j+1})) + \varepsilon_1 f(x_n^{i+\Delta i,j}) \\
&\quad + \varepsilon_2 f(x_n^{i+\Delta i,j+\Delta j}) \\
x_n^{i+m,j+m} &= x_n^{i,j},
\end{aligned}
\tag{7}
$$

a control parameter for the phase regulation and can be adapted in runtime.

The activated sites in the Coupling Layer can be additively coupled with other activated areas by additional terms and with different coupling parameters (e.g. $\varepsilon_1, \varepsilon_2$). The coupled sites represent four main links of each leg of the robot organism. The idea behind this approach is to synchronize all four links for achieving desired gaits (walking, trotting, galloping etc.). Depending on the structure of gait, links can be connected in parallel or crosswise, see Figure 18 (top), and can also be adapted dynamically to different situations. Therefore, additional coupling terms have been added to the equation 6, here exemplary a weak coupling ε_1 between links one (upper left) and two (lower left) and between links three (upper right) and four (lower right). The coupling parameter between links one and four and between two and three is ε_2 and is in this example much stronger than the ε_1

where Δi and Δj are distances between CML sites that represent the links. In order to analyze temporal effects of the phase propagation we need to pick the areas of interest and go through the time steps of the 2D map (Figure 15).

By choosing different values of coupling parameters we are able to synchronize the links for different locomotion patterns. As it can be exemplary observed in the Figure 18, legs can be synchronized for instance pairwise and in a cross-wise manner. It is of course not trivial to find always the suitable coupling parameter set, therefore evolutionary and learning approaches can run in parallel and learn it for achieving a good fitness.

As a conclusion to this section, we summarize our results. Using methods from non-linear dynamics and self-organization, we applied a CML-based approach for achieving synchronization between different limbs of a multi-robot organism. We analyzed serial as well as 2D coupled maps and also analyzed local and global impact of occurred disturbances.

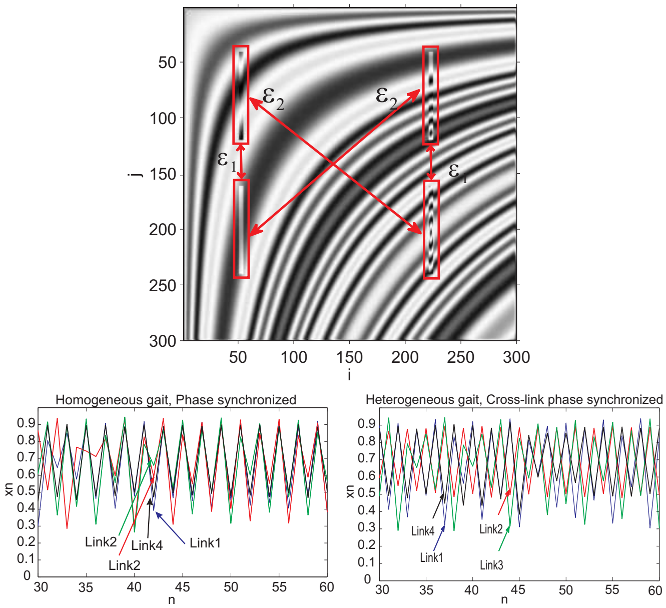


Figure 18. Time evolution of non-coupled (left image) and coupled (right image) sites in the activated areas for the four main links of the robot organisms. Coupling parameter are: $\varepsilon_1 = -0.01$ and $\varepsilon_2 = -0.001$. When system gets synchronized it can be observed that links one and four (blue and black lines) as well as the links two and three (red and green lines) get synchronized crosswise.

C. Fitness Driven Adaptability

Because it is very hard, to break down a desired behavior into the individual behavior of each robot in a swarm or in a multi-robot organism [17], we support the adaptation process with bio-inspired evolution. Especially, in unpredictable environments, when robots are able to (dis-)aggregate or modules may fail, pure classical approaches can perform suboptimal. Therefore, in addition to the adaptive mechanism based on self-organization from the previous section and artificial evolution of controller design, we adapt the robots by use of evolutionary concepts.

Figure 19 depicts the idea of the evolutionary concept. The key element is the genome, which contains the codified coupling matrix C_{ijk} and control parameters. This genome maps to the coupling of motors within the organism and thus to the behavior of the organism. Based on the previous section, the coupling strength can flow into the equations and extend the existing couplings. Detached from structural coupling, even functional coupling of not physically linked robots can evolve. For example the front left leg of a hexapod-like robot is not physically linked with the left back leg, but has to be synchronized in order to move. Even so, the strength of the coupling of two individuals can vary. Thereby, different strengths of the coupling lead to different behaviors. In order to adapt to a certain structure, which requires a non-trivial coupling, the strength of the couplings evolves.

A crucial point for evolution is the feedback from the

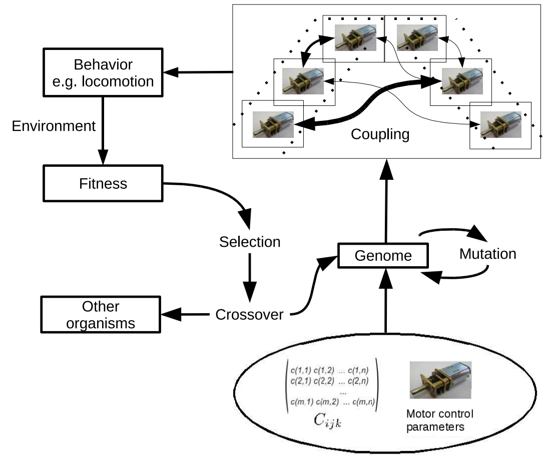


Figure 19. Evolutionary Concept: The genome contains the coupling matrix and parameter of the robot organism. This genome can be evolved over time and thus affects the behavior, respectively adapts the behavior. Through environmental and local feedback, the current fitness can be measured and optional crossover operators with other organisms can be applied.

environment. By evaluation of the current coupling structure in respect to a desired task (e.g. locomotion), a fitness value measures the performance of an organism in the current environment. In case of locomotion the fitness function could be influenced by multiple factors like power consumption, number of couplings (high number of coupling may lead to higher communication traffic) or velocity of an organism.

To cover each point potentially in the search space, mutational operators are used. This mutations can range from a single random change of a coupling entry to complete swapping of areas within the coupling matrix. In an environment with multiple organisms, we can use the evolutionary principle of cross-over. Two or more organisms can exchange their coupling matrices to each other. Depending on the fitness value of both parents, a total or a partial exchange of the genome can be done. The new structure can be either placed in one individual or in both.

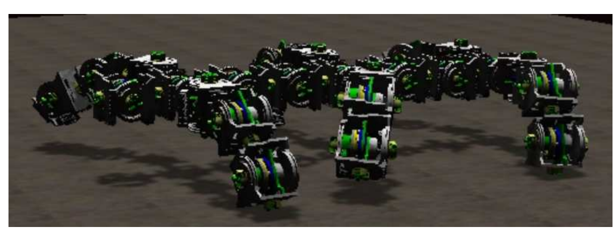


Figure 20. Exemplary hexapod organism in simulation.

For the design of controllers and the evaluation we use the Symbricator simulation [114]. Beside rings and caterpillar-like organisms we are interessted in legged organisms (like quadrupped or hexapod organisms, see Figure 20). The comparison of multiple (symmetrical and asymmetrical) shapes and the corresponding coupling matrices with the reached fitness values (in simulation) can lead to very efficient organisms and locomotion patterns. In the final stage, we want to pre-evolve the controllers in the simulation and deploy them afterwards to the real hardware. This will speed up the time for development and prevent hardware of critical damages. In a downstream step, the mechanism can adapt the robot online and onboard to unpredictable environments and changes during actual operating time.

D. Bound Self-concept and Structural Generation

Previous sections demonstrate examples of different adaptive processes on functional, structural and evolving levels. In this section we briefly introduce the generating level and self-concepts. As mentioned, adaptive and self-adaptive structures differ in two important points: adaptation uses environment for generating changes, whereas self-adaptation uses the self-concept for this purpose. Moreover, the self-adaptation is formulated in a more broad way than adaptation; to implement this, we need to integrate structural and functional generators into the system's architecture.

Thus, to utilize self-adaptive approach, we need to involve bound or unbound self-concepts and a generating mechanism. Several concrete examples of unbound self-concepts based on information theory are introduced in [1][10]. The unbound self-concept initiates unlimited (open-ended) growth of diversity and complexity; the treatment of this issue oversteps the framework of this paper. To explain the idea of self-concept for the structural generator, we focus on the bound case. The bound self-concept is invariant to adaptive processes. There are several mechanisms expressing such an invariant character of self-adaptation on the generating level: symmetries, templates and conservation laws, production, decomposition rules as well as self-reference. In this work we can briefly demonstrate the use of symmetries and symmetry breaking [115] for structural generation as well as ideas of developmental modularity [116] expressed in the form of "templates" for functional generation.

The most obvious way to generate well-scalable structural symmetries is to create a circulant [117] coupling

$$\underline{\underline{\mathbf{C}}} = \begin{pmatrix} T & 0 & c_1 \\ 0 & T & 0 \\ c_{n-1} & 0 & T \end{pmatrix}, \tag{8}$$

where $\underline{\underline{\mathbf{T}}}$ is a Toeplitz band matrix [118]

$$\underline{\underline{\mathbf{T}}} = \begin{pmatrix} c_0 & c_{n-1} & c_{n-2} \\ c_1 & c_0 & c_{n-1} \\ c_2 & c_1 & c_0 \end{pmatrix} \tag{9}$$

(taking into account dimensions n for $\underline{\underline{\mathbf{C}}}$ and for $\underline{\underline{\mathbf{T}}}$). The idea of introducing $\underline{\underline{\mathbf{T}}}$ consists in making topology and kinematics scalable to the size of this body. In this way, the basic building block is defined by circulant coupling $\underline{\underline{\mathbf{C}}} = circ(c_0, c_1, c_2, ..., c_{n-1})$. Well-known property of circulant coupling is a possibility of its diagonalization by the Fourier matrix

$$\underline{\underline{\mathbf{F}}} = \frac{1}{\sqrt{n}} \begin{pmatrix} 1 & 1 & 1 \\ 1 & w & w^2 \\ 1 & w^2 & w^4 \end{pmatrix}, \tag{10}$$

where

$$w = \cos\left(\frac{2\pi}{n}\right) + i \sin\left(\frac{2\pi}{n}\right). \tag{11}$$

The eigenvalues can be calculated as

$$\lambda_j = \sum_{i=1}^{n} c_i (w^{j-1})^{i-1}. \tag{12}$$

Maximal eigenvalue $\lambda_{max} = \lambda_1 = \sum_{i=1}^{n} c_i$, i.e., when circulant coupling has only a fixed number of c_i for any n, the stability and several other properties of $\underline{\underline{\mathbf{C}}}$ are invariant to the dimension of the whole system. Both, circulant and Toeplitz band matrices demonstrate ideas of invariances in the self-concept. From the view point of the group theory, $n \times n$ circulant can be viewed as a cyclic group Z/nZ of order n and can be generated by a generator g^n in Z/nZ. The generator g^n can represent a particular example of the bound self-concept, applied to generate scalable topological structures of an artificial organism.

To integrate symmetry breaking constrains into the topological self-concept, we can use the approach [119] in the form of [120]. Kiziltan and Milan in [120] defined four generators: R_f, C_f, which the flip first two rows/columns of a matrix and R_s, C_s, which shift the first row/column to the last position. For any generators, the notation $g \circ g = g^2$, (e.g. $R_f \circ R_f = R_f^2$) is used. Any two matrices are equivalent when they are obtained from each other by applying any of $R^n C^n$ generators, e.g. 3×3 coupling $\underline{\underline{\mathbf{C}}}$ has 36 symmetrical matrices. The idea of breaking a symmetry is to apply constrains, which order all symmetric objects, like the proposed lexicographical order [119].

Another concept behind self-generation are so-called templates. They are well-known in cognitive science [95] (also as "schemas" or "prototypes"), in topological research (in knot and braid theory) [96], as well as known as "frames" in AI community [97]. The idea of a template is to describe most general "stereotypical" properties or features of some common classes of situations/processes/objects. Concrete instance of a template can be reconstructed or generated by parametrization. There are several attempts to find an universal template, however it seems that different classes of solutions need different templates.

Since we are focusing on dynamic properties of collective actuation, we can assume each motor is driven by a periodic control. In this way a collective actuation represents a system of coupled oscillators with adaptive feedback, as e.g. described in [102]. As known, such systems possess self-adapting properties. Specific (desired, required) dynamic motion pattern can be generated when to parameterize the CML-driving-system with a specific set of control parameters as well as to provide a way to change these parameters, see Figure 21. Thus, we can map the problem of finding a dynamic template to the problem of finding such a bifurcation dynamics, which property reflects the needed changes. Speaking more technically, we are looking for universal

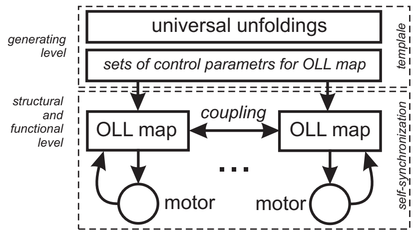


Figure 21. Template for collective locomotion.

unfolding [103]. Obviously, that universal unfolding together with parameter sets can be viewed as templates for collective actuation. Unfolding can be explained in the following way: let the normal forms of a local bifurcations be given by

$$\begin{aligned}\underline{\mathbf{q}}_{n+1} &= \underline{\underline{\Lambda}}_u(\{\alpha\},\{\beta\})\underline{\mathbf{q}}_n + \underline{\mathbf{g}}^{(2)}(\underline{\mathbf{q}}_n,\{\alpha\},\{\beta\}) + \\ &+ \quad ... + O(\underline{\mathbf{g}}^{(r+1)}), \qquad (13)\end{aligned}$$

where the term $\underline{\underline{\Lambda}}_u$ presents the diagonal matrix of eigenvalues, $\underline{\mathbf{g}}$ are the resonance terms, dependent on both $\{\alpha\}$ and $\{\beta\}$ and r is the determinancy order. Universal unfolding includes all possible perturbations of this normal form, which are equivalent to original bifurcation problem [103]. In this way, unfolding represents in some sense an invariance to perturbations. Finding universal unfolding allows defining the most general form of the desired dynamics, i.e. template. From the view point of dynamics, the universal unfolding can represent a bound self-concept, applied to rhythmic gait control.

VII. Conclusion

This paper has two main goals. The first goal was to demonstrate a common picture of adaptive processes and to represent key differences to self-adaptive mechanisms, which include bound and unbound self-concepts. Secondly, it was intended to exemplify these high-level concepts by one concrete example of collective locomotion in reconfigurable robotics.

The self-concept describes a goal of the system in some invariant form, such as symmetries, optimization principles, templates or information-based metrics. It can even generate an unlimited complexity and diversity, as proposed by von Neumann, in L-Systems as well as in self-referred dynamics [30]. It was argued that the origin of changes should be considered as the main difference between adaptation and self-adaption, and not the application of concrete approaches. Since self-adaptive mechanisms provide more degrees of freedom for modifications of behavior, functions or structures, corresponding generators should be integrated into the system's architecture. The self-adaptation and self-development overlap in several points; both concepts are driven in parallel by different communities. To provide consistency in logic of these notions, the self-development is considered to be more general ontogenetic mechanism, related to cognitive structures and their embodiment.

The introduced ideas are exemplified by the problem of collective locomotion in aggregated multi-robot organisms. Self-organizing and evolving adaptive mechanisms for the motion with obstacles have been considered. The synchronization of oscillators for motor drivers can reduce complexity of adaptive mechanisms. It was indicated that for performing further structural and functional changes without obstacles, another driving mechanism on the generating level should be used. Examples are given by symmetries and symmetry-breaking effects in structural matrices, templates or universal unfolding, which can represent a bound self-concept for functional and structural cases.

Several problems remain unsolved. Firstly, the coupling $\underline{\underline{\mathbf{C}}}$ in Figure 10 involves much more diverse structural and functional elements. It seems there is a complex dependency between structural and information couplings, which finally emerges a collective functionality. The whole framework around $\underline{\underline{\mathbf{C}}}$ requires more attention. Secondly, the structural self-concept based on symmetries of $\underline{\underline{\mathbf{C}}}$ can regulate morphodynamics of artificial organisms. However, it is completely unclear, how this concept can work with a more "high-level functionality", e.g. cognitive or hemostatic regulation. Finally, approaches in Sections VI-A – VI-D are only briefly sketched to indicate the used mechanisms. Experimental results for these sections are omitted, since these do not contribute to the main goal of this paper. It needs another work, which concentrates on these approaches and on a multitude of nonlinear effects appearing in them.

Acknowledgment

The "SYMBRION" project is funded by the European Commission within the work programme "Future and Emergent Technologies Proactive" under the grant agreement no. 216342. The "REPLICATOR" project is funded within the work programme "Cognitive Systems, Interaction, Robotics" under the grant agreement no. 216240. Additionally, we want to thank all members of projects for fruitful discussions.

References

[1] S. Kernbach, P. Levi, E. Meister, F. Schlachter, and O. Kernbach, "Towards self-adaptation of robot organisms with a high developmental plasticity," in *Proc. of the First IEEE International Conference on Adaptive and Self-adaptive Systems and Applications (IEEE ADAPTIVE 2009)*, J. Guerrero, Ed. Athens/Glyfada, Greece: IEEE Computer Society Press, 2009, pp. 180–187.

[2] K. Astrom and B. Wittenmark, *Adaptive Control*. Addison Wesley, 1989.

[3] M. Lungarella, G. Metta, R. Pfeifer, and G. Sandini, "Developmental robotics: a survey," *Connect. Sci.*, vol. 15, no. 4, pp. 151–190, 2003.

[4] S. Kernbach, E. Meister, O. Scholz, R. Humza, J. Liedke, L. Ricotti, J. Jemai, J. Havlik, and W. Liu, “Evolutionary robotics: The next-generation-platform for on-line and on-board artificial evolution,” in *Proc. of the IEEE Congress on Evolutionary Computation (IEEE CEC-2009)*, A. Tyrrell, Ed. Trondheim, Norway: IEEE Press, 2009.

[5] G. Baele, N. Bredeche, H. Haasdijk, S. Maere, N. Michiels, Y. Van de Peer, T. Schmickl, S. Ch., and R. Thenius, “Open-ended on-board evolutionary robotics for robot swarms,” *E-Commerce Technology, IEEE International Conference*, vol. 0, pp. 1123–1130, 2009.

[6] S. Kernbach, L. Ricotti, J. Liedke, P. Corradi, and M. Rothermel, “Study of macroscopic morphological features of symbiotic robotic organisms,” in *Proc. of the International Conference on Intelligent Robots and Systems (IROS-08), Workshop on Self-reconfigurable robots*, Nice, France, 2008, pp. 18–25.

[7] S. Kornienko, O. Kornienko, A. Nagarathinam, and P. Levi, “From real robot swarm to evolutionary multi-robot organism,” in *Proc. of the IEEE Congress on Evolutionary Computation (CEC-2007)*. Singapore: IEEE, 2007, pp. 1483–1490.

[8] SYMBRION, *SYMBRION: Symbiotic Evolutionary Robot Organisms, 7th Framework Programme Project No FP7-ICT-2007.8.2*. European Communities, 2008-2012.

[9] REPLICATOR, *REPLICATOR: Robotic Evolutionary Self-Programming and Self-Assembling Organisms, 7th Framework Programme Project No FP7-ICT-2007.2.1*. European Communities, 2008-2012.

[10] P. Levi and S. Kernbach, Eds., *Symbiotic Multi-Robot Organisms: Reliability, Adaptability, Evolution*. Springer-Verlag, 2010.

[11] S. Kernbach, *Structural Self-organization in Multi-Agents and Multi-Robotic Systems*. Logos Verlag, Berlin, 2008.

[12] A. Schore, *Affect Regulation And The Origin Of The Self: The Neurobiology of Emotional Development*. Lawrence Erlbaum Associates, Publishers, 1994.

[13] S. Kernbach, T. Schmickl, H. Hamann, J. Stradner, C. Schwarzer, F. Schlachter, A. Winfield, and R. Matthias, “Adaptive action selection mechanisms for evolutionary multimodular robotics,” in *Proc. of the 12th International Conference on the Synthesis and Simulation of Living Systems (AlifeXII)*. Odense, Denmark: MIT Press, 2010.

[14] S. Kernbach, “Concepts of symbiotic robot organisms,” in *Symbiotic Multi-Robot Organisms*, P. Levi and S. Kernbach, Eds. Berlin, Heidelberg, New York: Springer-Verlag, 2010, pp. 5–24.

[15] M. Asada, K. Hosoda, Y. Kuniyoshi, H. Ishiguro, T. Inui, Y. Yoshikawa, M. Ogino, and C. Yoshida, “Cognitive developmental robotics: A survey,” *IEEE Transactions on Autonomous Mental Development*, vol. 1, no. 1, pp. 12–34, April 2009.

[16] J. Spencer, M. Thomas, and J. McClelland, *Toward a Unified Theory of Development: Connectionism and Dynamic Systems Theory Re-Considered*. Oxford University Press, 2008.

[17] S. Nolfi and D. Floreano, *Evolutionary Robotics: The Biology, Intelligence, and Technology of Self-Organizing Machines*. Cambridge, MA. / London: The MIT Press, 2000.

[18] S. Elfwing, E. Uchibe, K. Doya, and H. Christensen, “Biologically inspired embodied evolution of survival,” in *Proc. of the 2008 IEEE Congress on Evolutionary Computation IEEE Congress on Evolutionary Computation*, Z. Michalewicz and R. G. Reynolds, Eds., vol. 3. Hong Kong: IEEE Press, June 2008, pp. 2210–2216.

[19] S. Ficici, R. Watson, and J. Pollack, “Embodied evolution: A response to challenges in evolutionary robotics,” in *Proc. of the Eighth European Workshop on Learning Robots*, J. L. Wyatt and J. Demiris, Eds., 1999, pp. 14–22. [Online]. Available: citeseer.ist.psu.edu/article/ficici99embodied.html

[20] C. Hartland and N. Bredeche, “Evolutionary robotics, anticipation and the reality gap,” in *ROBIO*. IEEE Press, 2006, pp. 1640–1645.

[21] S. Haroun Mahdavi and P. J. Bentley, “Innately adaptive robotics through embodied evolution,” *Auton. Robots*, vol. 20, no. 2, pp. 149–163, 2006.

[22] R. Frei, G. D. M. Serugendo, and J. Barata, “Designing self-organization for evolvable assembly systems,” in *Proc. of the Second IEEE International Conference on Self-Adaptive and Self-Organizing Systems (SASO-2008)*, 2008, pp. 97–106.

[23] A. Berns and S. Ghosh, “Dissecting self-* properties,” *Self-Adaptive and Self-Organizing Systems, International Conference on*, vol. 0, pp. 10–19, 2009.

[24] O. Brukman and S. Dolev, “Self-* programming: Run-time parallel control search for reflection box,” *Self-Adaptive and Self-Organizing Systems, International Conference on*, vol. 0, pp. 481–482, 2008.

[25] O. Babaoglu, M. Jelasity, A. Montresor, C. Fetzer, S. Leonardi, A. van Moorsel, and M. van Steen, *Self-star Properties in Complex Information Systems: Conceptual and Practical Foundations (Lecture Notes in Computer Science)*. Springer-Verlag New York, Inc., 2005.

[26] P. Boonma and J. Suzuki, “Exploring self-star properties in cognitive sensor networking,” in *Proc. of the International Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS-2008)*, 2008, pp. 36–43.

[27] E. Durfee, “Distributed problem solving and planning,” in *Multiagent Systems: A Modern Introduction to Distributed Artificial Intelligence*, G. Weiss, Ed. MIT Press, 1999, pp. 121–164.

[28] S. Kornienko, O. Kornienko, and P. Levi, “Multi-agent repairer of damaged process plans in manufacturing environment,” in *Proc. of the 8th Conf. on Intelligent Autonomous Systems (IAS-8), Amsterdam, NL*, 2004, pp. 485–494.

[29] B. Scassellati, "Building behaviors developmentally: A new formalism," in *Proc. of the Fifteenth National Conference on Artificial Intelligence (AAAI-98)*. AAAI Press, 1998.

[30] N. Kataoka and K. Kaneko, "Functional dynamics i: Articulation process," *Physica D*, vol. 138, pp. 255–250, 2000.

[31] J. R. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. Cambridge, MA: The MIT Press, 1992.

[32] D. Floreano and C. Mattiussi, *Bio-Inspired Artificial Intelligence: Theories, Methods, and Technologies*. MIT Press, Cambridge, Massachusetts, 2008.

[33] G. Williams, *Adaptation and Natural Selection*. Princeton: Princeton University Press, 1996.

[34] M. Conrad, *Adaptability: The Significance of Variability from Molecule to Ecosystem*. New York: Plenum Press, 1999.

[35] J. Gurvis and A. Calarco, *Adaptability: Responding Effectively to Change*. iiffer, 2007.

[36] H. Whitaker, "An adaptive system for control of the dynamics performances of aircraft and spacecraft," *Inst Aeronautical Services*, no. paper 59-100, 1959.

[37] P. Osbourne, H. Whitaker, and A. Kezer, "New developments in the design of model reference adaptive control systems," *Inst Aeronautical Services*, no. paper 61-39, 1961.

[38] B. Egardt, *Stability of Adaptive Controllers*, ser. Lecture Notes in Control and Information Sciences. Berlin: Springer-Verlag, 1979, vol. 20.

[39] B. Anderson, R. Bitmead, C. Johnson, P. Kokotovic, R. Kosut, I. Mareels, L. Praly, and B. Riedle, *Stability of Adaptive Systems: Passivity and Averaging Analysis*. Berlin: MIT Press, 1986.

[40] C. Rohrs, L. Valavani, M. Athans, and G. Stein, "Robustness of continuous-time adaptive control algorithms in the presence of unmodeled dynamics," *IEEE Transactions on Automatic Control*, vol. 30, no. 9, pp. 881–889, 1985.

[41] K. Narendra and A. Annaswamy, *Stable Adaptive Systems*. Prentice Hall, 1989.

[42] S. Sastry and M. Bodson, *Adaptive Control*. Prentice Hall, 1989.

[43] B. Anderson, "Failures of adaptive control theory and their resolution," *Communications in information and systems*, vol. 5, no. 1, pp. 1–20, 2005.

[44] R. Beer, *Intelligence as Adaptive Behaviour: An Experiment in Computational Neuroethology*. Academic Press, 1990.

[45] F. Keijzer, "Making decisions does not suffice for minimal cognition," *Adaptive Behavior*, vol. 11, no. 4, pp. 266–269, 2003.

[46] M. Mataric, "Situated robotics," *Encyclopedia of Cognitive Science, Nature Publishers Group, Macmillian Reference Ltd.*, 2002.

[47] R. Pfeifer, F. Iida, and G. Gomez, "Morphological computation for adaptive behavior and cognition," *International Congress Series*, no. 1292, 2006.

[48] L. Bull, M. Studley, A. Bagnall, and I. Whitley, "Learning classifier system ensembles with rule-sharing," *IEEE Transactions on Evolutionary Computation*, vol. 11, no. 4, pp. 496–502, August 2007.

[49] M. Butz, *Anticipatory Learning Classifier Systems*, ser. Genetic Algorithms and Evolutionary Computation. Kluwer Academic Publisher, 2002.

[50] M. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley-Interscience, April 1994. [Online]. Available: http://www.amazon.ca/exec/obidos/redirect?tag=citeulike09-20&path=ASIN/0471619779

[51] S. Kernbach, R. Thenius, O. Kernbach, and T. Schmickl, "Re-embodiment of honeybee aggregation behavior in artificial micro-robotic system," *Adaptive Behavior*, vol. 17, no. 3, pp. 237–259, 2009.

[52] G. Weiss, *Multiagent systems. A modern approach to distributed artificial intelligence*. MIT Press, 1999.

[53] D. Cliff, "Biologically-inspired computing approaches to cognitive systems: a partial tour of the literature," *Hewlett-Packard Company*, 2003.

[54] E. Alba and M. Tomassini, "Parallelism and evolutionary algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 5, pp. 443–462, 2002.

[55] SAP, *Adaptive Manufacturing: enabling the lean six sigma enterprises*. SAP, 2005.

[56] A. Ledeczi, G. Karsai, and T. Bapty, "Synthesis of self-adaptive software," in *Proc. of the IEEE Aerospace Conference 2000*. Big Sky, MT, 2000.

[57] B. Cheng, H. Giese, P. Inverardi, J. Magee, and R. de Lemos, *Software Engineering for Self-Adaptive Systems*. Dagstuhl Seminar 08031, 2008.

[58] W. R. Ashby, *Design for a brain: the origin of adaptive behavior*. New York,Wiley,, 1960.

[59] M. Wirsing, J.-P. Banâtre, M. M. Hölzl, and A. Rauschmayer, Eds., *Software-Intensive Systems and New Computing Paradigms - Challenges and Visions*, ser. Lecture Notes in Computer Science. Springer, 2008, vol. 5380.

[60] H. Colestock, *Industrial Robotics*. McGraw-Hill/TAB Electronics, 2008.

[61] K. Astrom, "Adaptive feedback control," *Proceedings of the IEEE*, vol. 75, no. 2, pp. 185–217, 1987.

[62] J. Vaario, “Modelling adaptive self-organization,” in *In Proceedings of the Fourth International Workshop on the Synthesis of Living Systems*, 1994, p. 314.

[63] T. M. Mitchell, *Machine Learning*. New York: McGraw-Hill, 1997.

[64] V. Chalam, *Adaptive Control Systems: Techniques And Applications*. CRC, 1987.

[65] D. Fogel, *Evolutionary Computation: Toward a New Philosophy of Machine Intelligence*. Piscataway, NJ: IEEE Press, 1995.

[66] K. J. Åström, “Design principles for self-tuning regulators,” in *Methods and Applications in Adaptive Control*, ser. Lecture Notes in Control and Information Sciences. Berlin: Springer-Verlag, Jan. 1980, no. 24, pp. 1–20.

[67] R. Kalman, “Design of a self-optimizing control system,” *Transactions of the ASME*, vol. 80, pp. 468–478, 1958.

[68] P. Hammond, *Theory of Self Adaptive Control Systems*. Plenum Press, 1966.

[69] A. Maslow, *Toward a Psychology of Being*. Hoboken, NJ: Wiley, 1998.

[70] K. McLean, M. Pasupathi, and J. Pals, “Selves creating stories creating selves: A process model of self-development,” *Personality and Social Psychology Review*, vol. 11, no. 3, pp. 262–278, 2007.

[71] L. Fausett, Ed., *Fundamentals of neural networks: architectures, algorithms, and applications*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1994.

[72] W.-M. Shen, M. Krivokon, H. Chiu, J. Everist, M. Rubenstein, and J. Venkatesh, “Multimode locomotion for reconfigurable robots,” *Autonomous Robots*, vol. 20, no. 2, pp. 165–177, 2006.

[73] H. Haken, *Synergetics: An introduction*. Berlin, Heidelberg: Springer-Verlag, 1977.

[74] J. Alvarenz-Ramírez, “Using nonlinear saturated feedback to control chaos: Hénon map,” *Physical Review*, vol. E48, no. 6, pp. 3165–3167, 1993.

[75] M. Basso, A. Evangelisti, R. Genesio, and A. Tesi, “On bifurcation control in time delay feedback systems,” *Int. J. Bifurcation and Chaos*, vol. 8, no. 4, pp. 713–721, 1998.

[76] A. Kolesnikov, *Synergetic control theory (in russian)*. Taganrog: Taganrog state university, 1994.

[77] T. Bäck, “Introduction to the special issue: Self-adaptation,” *Evol. Comput.*, vol. 9, no. 2, pp. 3–4, 2001.

[78] H.-G. Beyer, “Toward a theory of evolution strategies: Self-adaptation,” *Evol. Comput.*, vol. 3, no. 3, pp. 311–347, 1995.

[79] M. Ibrahim, R. Anthony, T. Eymann, A. Taleb-Bendiab, and L. Gruenwald, “Exploring adaptation & self-adaptation in autonomic computing systems,” *Database and Expert Systems Applications, International Workshop on*, vol. 0, pp. 129–138, 2006.

[80] A. Avstreih, “The emerging self: Psychoanalytic concepts of self development and their implications for dance therapy,” *American Journal of Dance Therapy*, vol. 4, no. 2, pp. 21–32, 1981.

[81] A. Bonarini, A. Lazaric, M. Restelli, and P. Vitali, “Self-development framework for reinforcement learning agents,” in *Proc. of the 5th International Conference on Development and Learning (ICDL06)*, Bloomington, USA, 2006.

[82] P.-Y. Oudeyer and F. Kaplan, “Intelligent adaptive curiosity: a source of self-development,” in *Lund University Cognitive Studies*, 2004, pp. 127–130.

[83] J. Pujol, P. Vendrell, C. Junque, J. Marti’-Vilalta, and A. Capdevila, “When does human brain development end? evidence of corpus callosum growth up to adulthood,” *Annals of Neurology*, vol. 34, no. 1, pp. 71–75, 1994.

[84] T. Lee and A. Peterson, “Adaptive vector quantization using a self-development neural network,” *IEEE Journal on Selected Areas in Communications*, vol. 8, no. 8, pp. 1458–1471, Oct 1990.

[85] S. Rasmussen, L. Chen, D. Deamer, D. Krakauer, N. Packard, P. Stadler, and M. Bedau, “Evolution: Transitions from nonliving to living matter,” *Science*, vol. 303, no. 5660, pp. 963–965, 2004.

[86] J. von Neumann, *Theory of Self-Reproducing Automata*. Champaign, IL, USA: University of Illinois Press, 1966.

[87] C. Waddington, “Paradigm for an evolutionary process,” in *Towards a Theoretical Biology*, C. Waddington, Ed. Edinburgh University Press, 1969, vol. 2, pp. 106–128.

[88] K. Ruiz-Mirazo, J. Umerez, and A. Moreno, “Enabling conditions for ”open-ended evolution”,” *Biology & Philosophy*, vol. 23, pp. 67–85, 2008/01 2008.

[89] C. Maley, “Four steps toward open-ended evolution,” in *Proc. of the Genetic and Evolutionary Computation Conference (GECCO-99)*. Morgan Kaufmann, 1999, pp. 1336–1343.

[90] B. Rosslenbroich, “The theory of increasing autonomy in evolution: a proposal for understanding macroevolutionary innovations,” *Biology & Philosophy*, 2009.

[91] L. Spector, J. Klein, and M. Feinstein, “Division blocks and the open-ended evolution of development, form, and behavior,” in *Proc. of the 9th annual conference on Genetic and evolutionary computation (GECCO-07)*. London, England: ACM, 2007, pp. 316–323.

[92] R. Standish, “Open-ended artificial evolution,” *Int. journal of computational intelligence and application*, vol. 3, p. 167, 2003. [Online]. Available: http://www.citebase.org/abstract?id=oai:arXiv.org:nlin/0210027

[93] J. McCormack, “Interactive evolution of l-system grammars for computer graphics modelling,” in *Complex systems: from biology to computation*, D. Green and T. Bossomaier, Eds. IOS Press, 1993, pp. 118–130.

[94] K. Kouptsov, “Production-rule complexity of recursive structures,” in *Unifying Themes in Complex Systems IV, Proc. of the Fourth International Conference on Complex Systems*, A. Minai and Y. Bar-Yam, Eds. Springer-Verlag Berlin, Heidelberg, 2008, pp. 149–157.

[95] F. Gobet and H. Simon, “Templates in chess memory: A mechanism for recalling several boards,” *Cognirtive Psychology*, vol. 31, pp. 1–40, 1996.

[96] J. Birman, “Braids, knots and contact structures,” 2004. [Online]. Available: http://www.citebase.org/abstract?id=oai:arXiv.org:math/0403489

[97] M. Minsky, “Frame-system theory,” in *Thinking. Readings in cognitive science*, P. N. Johnson-Laird and P. C. Wason, Eds. Cambridge: Cambridge University Press, 1977.

[98] R. Manseur and K. L. Doty, “A complete kinematic analysis of four-revolute-axis robot manipulators,” *ASME Mechanisms and Machines*, vol. 27, no. 5, pp. 575–586, 1992.

[99] D. Floreano, P. Husbands, and S. Nolfi, “Evolutionary Robotics,” in *Handbook of Robotics*. Berlin: Springer-Verlag, 2008.

[100] G. Endo, J. Morimoto, T. Matsubara, J. Nakanishi, and G. Cheng, “Learning cpg-based biped locomotion with a policy gradient method: Application to a humanoid robot,” *Int. J. Rob. Res.*, vol. 27, no. 2, pp. 213–228, 2008.

[101] K. Kaneko., *Theory and application of coupled map lattices*. John Willey & Sons, Inc., 1993.

[102] Y. Kuniyoshi and S. Sangawa, “Early motor development from partially ordered neural-body dynamics: experiments with a cortico-spinal-musculo-skeletal model,” *Biological Cybernetics*, vol. 95, no. 6, pp. 589–605, 2006.

[103] M. Golubitsky and D. Schaeffer, *Singularities and groups in bifurcation theory*. New York: Springer-Verlag, 1985, vol. I.

[104] Y. Gu, M. Tung, J.-M. Yuan, D. Feng, and L. Narducci, “Crises and hysteresis in coupled logistic map,” *Phys. Rev. Letters*, vol. 25, no. 9, pp. 701–704, 1984.

[105] Y. Maistrenko, V. Maistrenko, and A. Popovich, “Transverse instability and riddled basin in a system of two coupled logistic maps,” *Phys. Rev. E*, vol. 57, no. 3, pp. 2713–2724, 1998.

[106] O. Kernbach, *The synergetic approach towards analysing and controlling the collective phenomena in multi-agents systems, PhD thesis*. University of Stuttgart: to be published, 2010.

[107] Y. G.Taga and H. Shimizu, “Self-organized control of bipedal locomotion by neural oscillators in unpredictable environment,” *Biological Cybernetics*, vol. 65, pp. 147–159, 1991.

[108] S. Y. S. Yonekura and Y. Kawaguchi, “Adaptive body use of multipede driven by coupled neural oscillators,” in *18th International Conference on Artificial Reality and Telexistence (ICAT 2008)*, 2008.

[109] K. Pyragas, “Control of chaos via extended delay feedback,” *Physical Letters A*, vol. 206, pp. 323–330, 1995.

[110] K. Konishi and H. Kokame, “Time-delay-induced amplitude death in chaotic map lattices and its avoiding control,” *Physical Letters A*, vol. 366, pp. 585–590, 2007.

[111] M. L. A. Pitti and Y. Kuniyoshi, “Quantification of emergent behaviors induced by feedback resonance of chaos,” 2005.

[112] ——, “Generating spaciotemporal joint torque patterns from dynamical synchronization of distributed patterns generators,” *frontiers in neurorobotics*, vol. 3, 2009.

[113] M. Mehta and S. Sinha, “Asynchronous updating of coupled map leads to synchronization,” *Chaos*, vol. 10, 2004.

[114] L. Winkler and H. Wörn, “Symbricator3d - a distributed simulation environment for modular robots,” in *Intelligent Robotics and Applications*, ser. Lecture Notes in Computer Science, vol. 5928/2009. Springer Berlin / Heidelberg, 2009, pp. 1266–1277.

[115] B. Johnston and F. Richman, *Numbers and Symmetry: An Introduction to Algebra*. CRC-Press, 1997.

[116] A. Karmiloff-Smith, *Beyond Modularity: A Developmental Perspective on Cognitive Science*. MIT Press, 1995.

[117] P. Davis, *Circulant matrices*. John Willey & Sons, 1979.

[118] A. Böttcher and B. Silbermann, *Introduction to large truncated Toeplitz matrices*. New York: Springer-Verlag, 1999.

[119] P. Flener, A. Frisch, B. Hnich, Z. Kiziltan, I. Miguel, J. Pearson, and T. Walsh, “Breaking row and column symmetries in matrix models,” in *Proc. of the 8th International Conference on Principles and Practice of Constraint Programming (CP-02)*. London, UK: Springer-Verlag, 2002, pp. 462–476.

[120] Z. Kiziltan and M. Milano, “Group-graphs associated with row and column symmetries of matrix models: Some observations,” in *Proc. of the Second International Workshop on Symmetry in Constraint Satisfaction Problems (SymCon-02)*, Ithaca, NY, USA, 2002.

Distributed Smart Spaces M3 Platform for Building Professional Social Networks with Seamless PCs and Mobile Access

Sergey Balandin, Ian Oliver, and Sergey Boldyrev
Ubiquitous Architectures team, Helsinki lab
Nokia Research Center, Nokia
Itamerenkatu 11-13, 00180 Helsinki, Finland
{Sergey.Balandin, Ian.Oliver, Sergey.Boldyrev}@nokia.com

Abstract — this paper proposes the social network solution that has been designed to fit equally well for use at mobile devices, PCs and even other types of consumer electronics. This solution is targeted to become a core element of the personal knowledge space and the design has be been done on top of M3 smart space. Target of this paper is to improve and to expand the understanding of the Smart Spaces concept by the R&D community. Through the identification of key properties based on an analysis of evolving trends brought to us by the great convergence in the ICT industry. Based on that we show how this trend will affect adoption of Smart Spaces. It is especially important to understand how Smart Spaces can change the whole services ecosystem and the role that mobile and other types of user surrounding devices will play. The Smart Space concept can be described as a permanent robust infrastructure to store and retrieve information of various types from a broad spectrum of different environment participants. This concept can provide better user experience by allowing a user to bring flexibly to the new devices and access all the information in the multi device system from any device. Based on that the resulting social network solution will be able to pre- and post-process the collected information and perform efficient reasoning over and organization of the data. Another key advantage of the proposed social network is that it is primary targeted to support professional social communications within a geo-distributed teams working on the same project. For that the solution proposes a flexible and easily extendable set of additional services, such as a variety of conference call and virtual meeting services with logging service, sharable whiteboard, automatic maintenance of action point lists and calendars and so on. As a result the service provides users with completely new experience.

Keywords-social networks for mobile devices, smart spaces; M3; supporting distributed R&D projects.

I. Introduction

This paper is based on the conference paper presented at Ubicomm 2009 conference [1]. Social networks are attracting more and more Internet users and soon the absolute majority of the Internet users will have a membership at least in one social network and so social networks become the most demanded type of service in the Internet. Also we have to take into account that in Japan already in 2005 the number of mobile users exceeded PC connections [2] and starting from second half of 2008, the world-wide number of new users connected to Internet through mobile devices is growing faster than users with a classical access from the PCs. This creates a strong demand for a social network solution that equally well fits for use from PCs and mobile devices.

At the same time the new information age mandates development of a permanent robust infrastructure to store and retrieve information of various types from a broad spectrum of different network participants. The environments that fulfill such requirement are called "smart spaces" and assume presence of a number of devices that use shared view of the resources and services. The smart spaces can provide better user experience by allowing a user to bring flexibly to the new devices and access all the information in the multi device system from any device. The challenge of providing consistent information in a smart space is that the resources are distributed over several physical devices and the information consumption is not always happening at the same device where the information is located.

Recently we observed an emergence of the smart space technologies, which are truly mobile by the definition of their nature and allow equally efficient design of applications for PC and mobile devices. An example of such smart spaces platform is Multipart, Multidevice and Multivendor (M3) platform [3][4][5], which has been developed by a consortium of companies lead by Nokia. It gives the user a great flexibility in design and easy to use tools for personalizing the user account and access application as much as user wants. We believe that by designing social network platform on top of M3 we will provide a user with the new truly mobile social network experience and allow integration of social network's information flow with other information flows going via the mobile device and PCs of the user, while keeping all personal data under full physical control of the owner.

Despite a huge number of mobile Internet users, nowadays there is no widely known social network solution that has been specifically designed for mobile use. So when nowadays the mobile user wants to be socially networked he/she cannot get this service or in the best case is forced to use the PC client adopted for mobile device. As a consequence this creates a lot of problems and inconveniences for the user.

The most visible problem is inefficient User Interface (UI), which is usually the scaled down version of PC client UI with some functional cuts comparing to the PC version.

Moreover this UI cannot take into account personal preferences and restrictions of the user and as a result most social network users use the mobile client rarely and only when they do not have other alternatives, which definitely is an unacceptable situation. It would be great if the user gets a possibility to easily build personalized client UI out of the provided library of blocks and when needed easily to make even completely new blocks or blocks with the inherited functionality.

Some other key problems are placed deeply in the architectural principles of the currently available social network solutions. All of them have been designed as stand alone applications for usage on PC, with client-server principles in mind and data archive stored as a repository of short text messages. It has resulted in the monolithic architecture of the social network client, platform dependency, lack of flexibility in selecting location of the data repository, and over-complicated (often even fully forbidding) schemes of information sharing, reasoning over and joint use of the collected data in cooperation with other services.

For example, nowadays people share a lot of personal information via social networks and the fact that this information belongs to the social network host makes more and more people to consider limiting their activities in social networks, even if they see a value of the service, but at some point the privacy issues start to dominate. Another pity fact is that the huge amount of personal and community information is stored in the system without efficient use, while by applying to it methods of automatic data reasoning would open door for provision of a number of additional services.

In order to further support growth of social network services and allow their equally efficient use from all types of devices, a new solution has to be proposed. The next section provides general description of Smart-M3 smart space reference model. After that we give the basic definition of the new social networks solution, which is followed by a description of one possible implementation of it on top of Smart-M3. At the end of paper we provide a set of key conclusions of this study and a list of references.

II. M3 Smart Space Reference Model

In the book by Diane Cook and Sajal Das the following formal definition of Smart Spaces is given: "Smart Space is able to acquire and apply knowledge about its environment and to adapt to its inhabitants in order to improve their experience in that environment" [6][7]. This definition assumes continues interaction of the user with the surrounding environment that is targeted in continuous adaptation of the services to the current needs of the user. This interaction is enabled by sensing functionality that gathers information about the space and the user; adaptation functionality for reacting to the detected changes; and effecting functionality for changing the surrounding space to benefit the user. Based on the definition the main focus of Smart Spaces is on the user. The general view of the Smart Spaces hierarchy is depicted by Figure 1.

Figure 1. Hierarchical layers of Smart Spaces with user in the center.

Obvious key concepts for any Smart Space are mobility, distribution and context awareness. These are addressed by the recent advances in wireless networking technologies as well as processing and storage capabilities, which have moved mobile and consumer electronics devices beyond their traditional areas of applications and allow their use for a broader scope of services. The significant computing power and high-speed data connections of the modern mobile devices allow them to become information processing and communication hubs that perform rather complex computations locally and distribute the results. This lets multiple devices interact with each other and form ad-hoc dynamic, distributed computation platforms. Together, they form a space where via a number of wireless technologies the users can access a huge variety of services. Similarly, existing and future services form spaces that cater for a variety of needs ranging from browsing to interactive video conversations. These services surround the user all the time and have access to large amounts of data. Over time they can learn the users' needs and personal preferences, making it possible to build even more advanced services that proactively predict those needs and propose valuable services in the given environment before the users realize it themselves. These layers, each of which can utilize a number of technologies form a smart environment (Smart Space). A further important aspect is that Smart Spaces improve the interaction between users and their physical environments, allowing more efficient consumption of available resources such as energy. Examples of the existing smart space solutions can be found in [8][9].

Based on the analysis of existing smart space environments one can notice that the essential features demanded from all these systems is that they should provide permanent robust infrastructure to store and retrieve information of different kinds from the multitude of different types of the environmental actors. For example one can see the personal space as a framework to interact, manipulate and share information represented using own local semantics.

For the later discussed class of the smart space solutions the following assumptions have been made.

- The key concepts for smart space are mobility, distribution and context awareness.
- The user interacts with a space using multiple agents, where each agent implements certain atom of functionality and services and applications are built as a combined functionality of one or more agents.
- The core of space is information storage A, on top of which reasoning and deductive closure methods create corpus R(A).
- The space information storage and agents are distributed and located on a multitude of devices.
- The user can simultaneously interact with many discrete spaces.

Based on the above defined principles the high-level definition of the target smart space reference model can be illustrated by Figure 2.

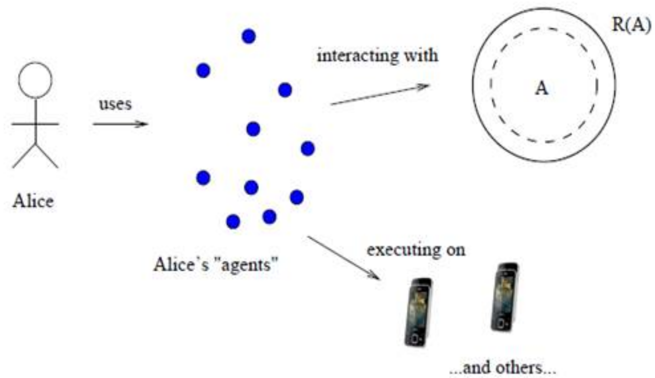


Figure 2. The reference model of Smart-M3 smart space.

On top of this reference model the Smart-M3 solution has been created. Smart-M3 is an interoperability platform that provides mechanisms to share information expressed using RDF. The platform consists of a Semantic Information Broker (SIB) that stores the information, and Knowledge Processors (KP) that acts as agents and can insert, remove, query and subscribe to information. Figure 3 shows a simple functional architecture of Smart-M3.

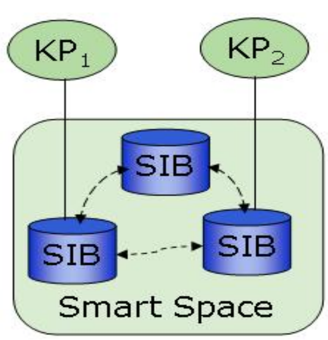


Figure 3. The model of simple functional architecture of Smart-M3.

The information is stored in Smart-M3 according to some standardized or otherwise agreed ontology. But it is important to note that Smart-M3 doesn't require global ontological alignment, instead only local alignment is required. Information sharing helps the participating KPs in gathering relevant context information, which leads to more efficient operation and allows innovative multi-device use cases. During the lifetime SIBs goes through multiple cycles of merge and split operations and the user at the same time could participate in multiples spaces. The physical representation of Smart-M3 reference model is shown by Figure 4.

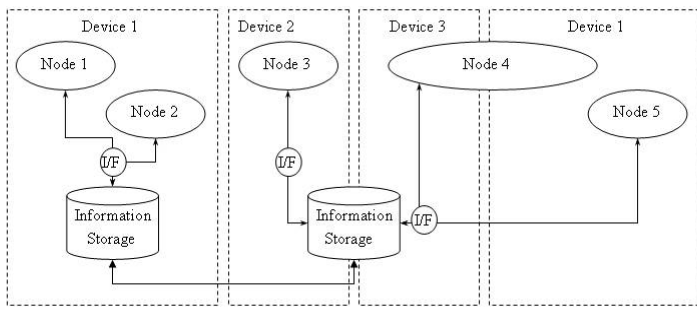


Figure 4. Physical representstion of the reference model of Smart-M3.

Where:

I/F is an *interface* - that provides information exchange between the nodes and information storages. The interface is considered to be fully reliable and does not create additional delay and energy overheads. In this reference model the interface performs a technical function of connecting nodes to information storages. It does not implement logical functions and does not affect information transfer costs. For this reason the interface is not considered in the mathematical model.

Information storages - are also logical units that can be located at several devices and several information storages can be located on one device.

Nodes - are logical elements capable to perform certain actions. One node can be located at several physical devices and several nodes can be located at one device.

Information is described by *information units* (IU) - represented as logical expressions: "subject"-"predicate"-"object" = [true | false], where *subject* is an actor (human or node that performs certain actions), *predicate* is an action that is being performed or supposed to be performed (e.g., "playing music") and *object* is what the action is performed with (e.g., a song being played). The nodes have predefined *behavior rules* defining their actions in line with the received information units.

Based on that the Smart-M3 smart space can consist of the multiple, individual autonomous spaces, with information distributed over multiple devices, including embedded solutions, OVI, PC, mobile, etc., as is shown in Figure 5

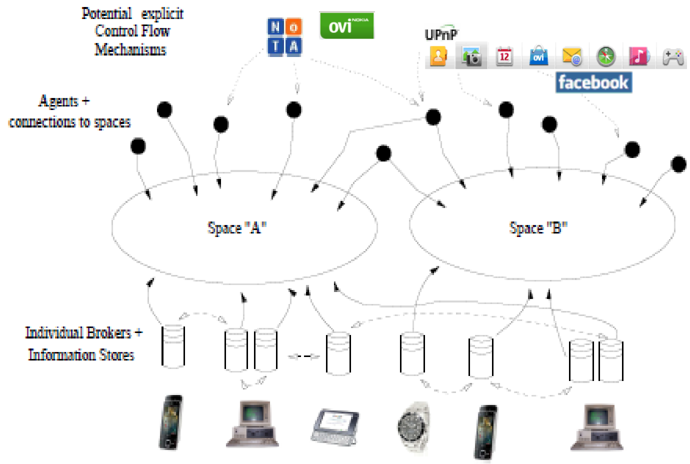


Figure 5. The extended reference model of Smart-M3 smart space.

In a nutshell, the Smart-M3 smart space consists of one or several SIBs. The rules of information usage (user applications) are implemented in KP connected to the smart space via SIBs. The SIBs are responsible for storing smart space information and its sharing. As soon as an information unit becomes available for the SIB, it becomes available for every KP.

As it is mentioned earlier, Knowledge Processors (KPs) are responsible for information processing. In fact the Smart-M3 applications emerge as a combination of one or multiple KPs connected to the corresponding information in the SIBs as illustrated by Figure 6.

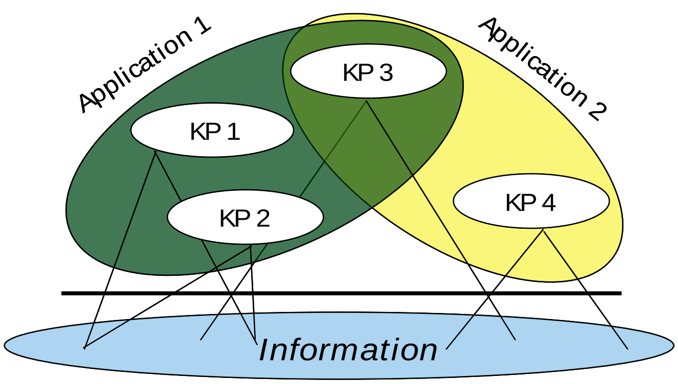


Figure 6. Smart-M3 applications emerge as a combination of KPs connected to SIBs.

It is important to note that the same KP might contribute to the functionality of more than one application. This architecture allows developing new and multiplying existing application with functionality needed for solving current user needs. Also this solution gives developers the great flexibility for deep customization of the application with the minimum overhead, as in most cases the functionality of question can be achieved by changing only one or few KPs, without touching the rest of the application. Altogether the above described features provides the great basis for developing efficient and flexible solutions for different types of user devices, which is exactly what is needed when one think about social network solutions.

III. Description of the Proposed Solution

To start let's see what are the main differentiators and advantages of building social network on top of Smart-M3 comparing to the traditional social network architectures:

1) The internal structure of social network node can be illustrated by the scheme presented in Figure 7.

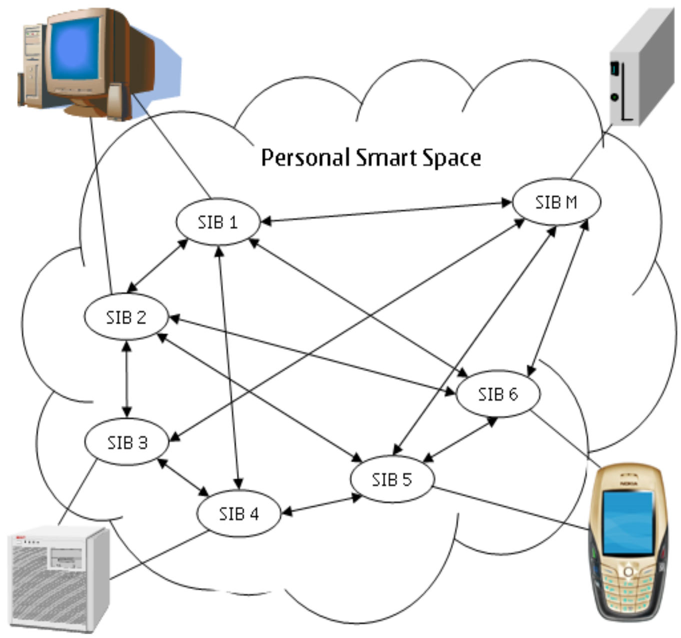


Figure 7. Organization of personal smart space: structure of the social network node.

The proposed social network solution is organized based on peer-to-peer principle (note that here peer-to-peer only refers to the general architectural principle, as oppose to client-server architecture), when all personal data is stored in the space that is under physical control of the owners. So every peer node can be seen as a personal space of the user, which could be physically distributed over a number of devices of different types, e.g., PCs, mobile devices, public and personal data storages, etc. The key feature of each node is that the underlying platform provides a user with the same access capabilities to all elements of the personal data from any device that he/she entered to the personal Smart Space.

2) The social network solution is based on use of Semantic Information Brokers (SIB), which integrates it with other services and information available in the SIB and makes it equally efficient for use from PC and mobile device. Every user has his/her own space, called Personal Smart Space, as is shown in Figure 7. When users decide to become friends in the social network, they not just exchange view/edit access rights to the information pages of each other, but allow real sharing of a certain part of personal space with the new friend peer. This way a group Social Space is created as a merge of certain parts of the personal spaces of group members, where the group members could

be individuals and other already existing user groups, as it is illustrated in Figure 8.

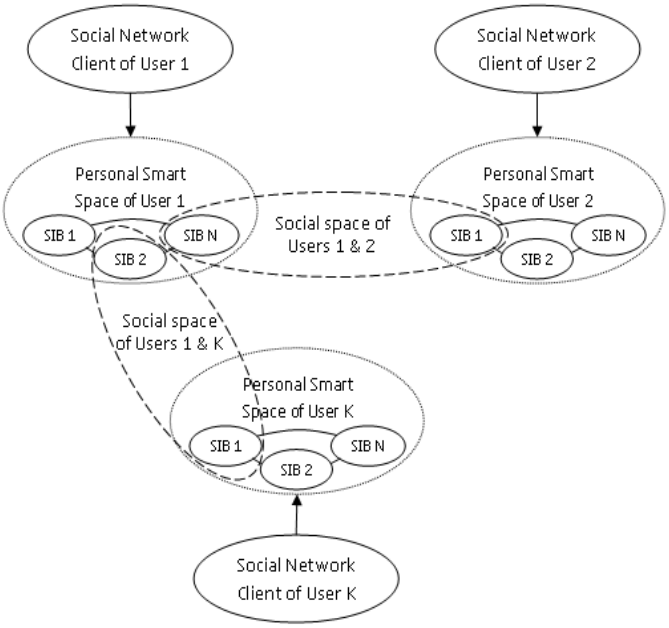


Figure 8. Building social spaces of the friend users as a merge of their personal smart spaces.

Within Smart-M3 definitions the new Social Space is created as a merge of projections of the information sets that the users have decided to share with a given peer (on Figure 4 these spaces are shown by dashed lined ovals). Note that the proposed solution allows user to differentiate friends as much as he/she wants, not restricted by the rules defined by the social network's host. The user can share a lot of personal data with best friends, work related materials with colleagues and the sets of shared information could be completely isolated from each other, or have partial or full overlapping. Also the social space can be expanded to as many users as needed and so the larger group's social space will be created, or alternatively the peers' social spaces can enter into a larger group's space still preserving autonomy.

3) The proposed solution provides great flexibility in defining social network client. The client application is a combination of the M3 applications (hereafter, knowledge processor or KP), which can be seen as a bricks for providing users with the exact functionality that is demanded. Remember that physically any KPs can be located at any device. It is only demanded that they all are connected to the personal social space, i.e., to the user node. Such architecture gives great flexibility for adoption to the actual capabilities of the host device, personalization of the social network client, which includes functional and UI parts. The basic architecture and work principles of the resulting client application can be illustrated by Figure 9.

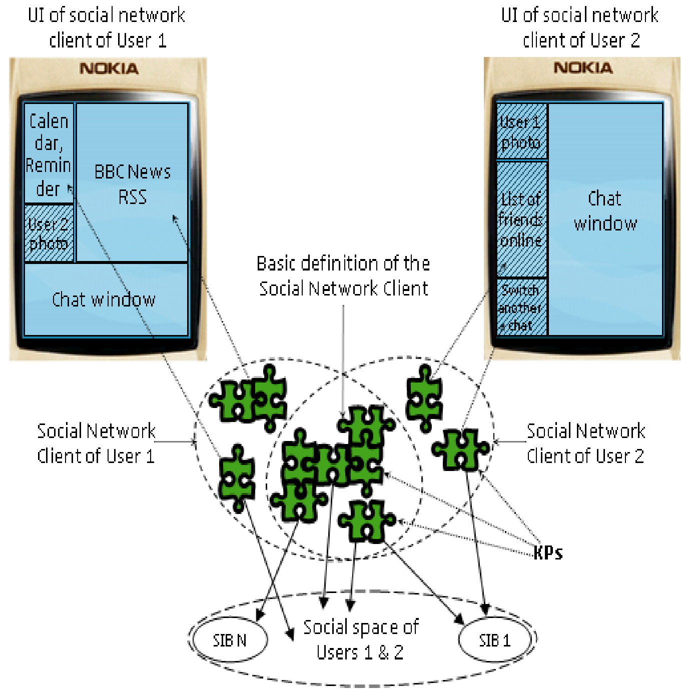


Figure 9. Architecture of the social network client application.

For example, the set of basic configuration definitions of the social network client application can be provided as a part of default distribution package. Then on top of this basic functionality every user can personalize the client by adding KPs that customize functionality and UI. For examples, the additional KPs can be visible on UI, as it is illustrated on Figure 9 by KPs that implement and output "Calendar, Reminder" and "BBC News RSS" functionality for User 1 and "List of friends online" and "switch to another active chat" for User 2. Other KPs can be invisible and used for performing some background actions with the collected data. For example, the user can add KP for performing reasoning over information extracted from SMS and chat engine of the social network, to define what social events in the town might be of a mutual interest for the selected group of friends. Another KP might take care of the smart delivery of important messages to the selected friends, i.e., taking into account all user preferences, especially those that have not been explicitly specified, but obtained from reasoning over user's behavior using all devices and applications in Smart-M3, and so on.

Taking into account the above described advantages of Smart-M3 as a platform for social network we decided to make the first social network application of this type as a framework for establishing professional social networks and supporting all-time cooperation in the distributed R&D projects. This idea comes to our mind, while looking for a solution to facilitate joint work in multi-site projects, we have discovered that none of the existing solutions cover this niche. In fact all the basic modules as in place, e.g., VOIP and shared data repositories, social networks of friends, like LinkedIn and Facebook, but you will not be able to find efficient and easy-to-use framework that bridges all these modules for not only getting all these services via the same application, but most importantly to benefit from data flows and capabilities of all involved modules. Nowadays in most cases such bridging is performed by the user, but the automated intelligent solution is strongly demanded. The required solution should be easily accessible from all user devices, e.g., PCs and mobile devices and the state of project activities including UI and database settings must follow user's migration between the devices. Also a lot of new services such as short voice messages, shared touch-screen whiteboard for participants of teleconference and so on, have to be introduced.

An important feature of target system is that by introducing additional KPs the user should be able to link accounts and services of other social networks into the created framework. So the developed social network client (similarly as it is illustrated in Figure 6 for BBC News RRS) can be an interface and the data processing module for all social networks where the user is registered. As a result the user can use one application and smoothly move his/her social life to the most personalized and appropriate form.

IV. Implementation

As it is described in the previous section, the Social Spaces are defined as a merge of the projections of information sets that the users have decided to share with the given sets of peer nodes. The main consequence of it is that the solution does not require social network-wide definition of the data ontology. Instead every user can stay within the scope of the personal smart space ontology definition and the required commonality will be achieved at the level of joint use of the projections.

The social network client is defined as a Smart-M3 application and consists of a number of knowledge processors that perform designated operations over information in the personal and groups' social spaces and allow access to the service from PCs, and Symbian and MAEMO mobile devices. In this chapter we define only the basic set of KPs and provide high level description of the corresponding operational sequence diagrams required for the service. But as it has been discussed before, one of the key benefits of the proposed solution is that any user will be able to personalize social network client as much as he/she wants and correspondingly the number and variety of the related KPs will grow with the time and even the below described basic set is a subject for continuous updates and modifications.

And before diving into details of the social network solution implementation it is important to say a few words about the underlying development platform. Currently available Smart-M3 open source release contains KP APIs for C/GLib, Python, and Qt/C++. The KPs can be connected to SIB by using either TCP/IP or NoTA H_IN protocol. Eventually more support for different programming languages and connectivity mechanisms are in the to-be-added plan. There is also an ontology library generator for C/GLib API that allows developers to program using ontology concepts instead of using Smart-M3 basic operations and RDF. Ontology library generators for other languages are also forthcoming.

The most recent version of Smart-M3 M3/Sedvice services development platform is available to all developers under open license [10].

The default package of the developed Smart-M3 social network client consists of the following Knowledge Processors:

1) *Manager of Social Spaces*. This KP is responsible for creating new social spaces for user groups and maintaining the list of available social spaces. This is a key functionality as it allows any user create personalized hierarchy of the social spaces around. For example, one can have space for family, elementary school classmates, university classmates, work colleagues and so on. It is clear that these spaces can have certain overlapping, as the same person could be your university classmate and wife. Also when a new peer is identified the user needs to define whether to add it to the existing social space, or create a new space from the scratch or as an extension of another already existing space. On top of this KP allows to search the space by using either peer-to-peer principle or via special "open communities formation" registration servers, which contain only user ID details and basic details how to contact user, but all actual content is on the user side. The example implementation of this KP is presented in the sequence diagram format in Figure 10.

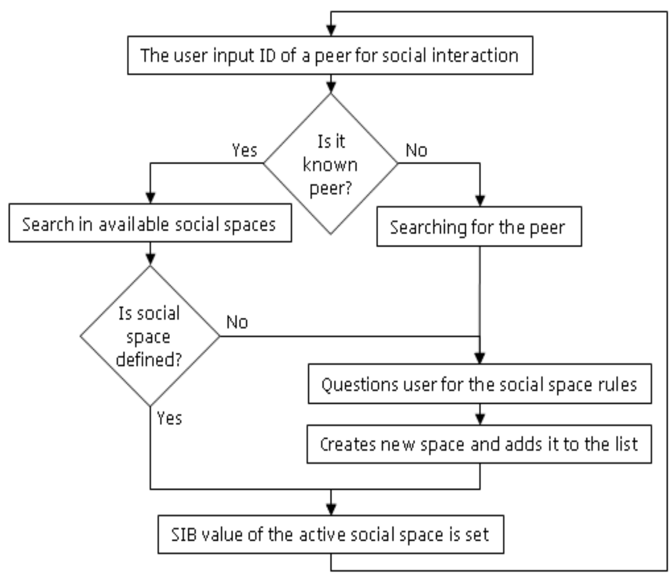


Figure 10. Sequence diagram of the Manager of Social Spaces.

2) *Social Space communicator*. This KP defines all basic queries, subscription methods, handling of collisions and information access rights in a given social space for each person and group. It also provides the user with a set of basic "must to have" services, e.g., the chat client, sharable calendar and organizer, file exchange, image sharing, white board, etc. In fact this client defines the basic common set of services that should be available for all members of the smart space and guaranties that the basic communication mechanisms are in place. One can see it as an internal translator KP that transforms complex functionalities of

various services into a sequence of primitive queries. The sequence diagram for the corresponding KP implementation is presented in Figure 11.

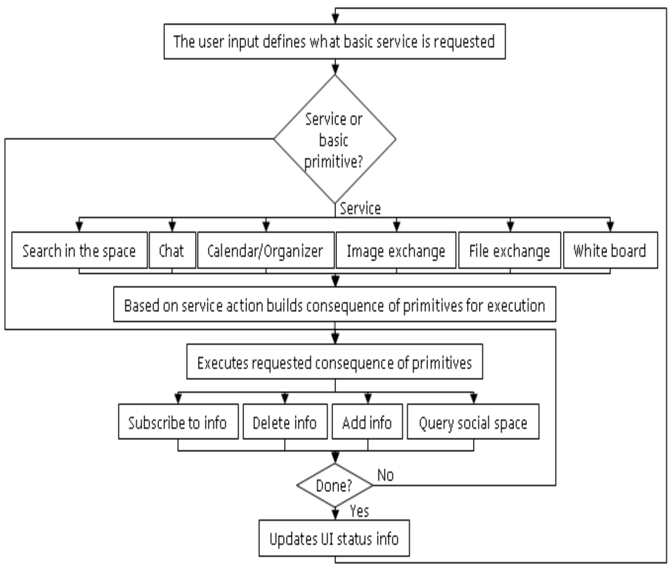


Figure 11. Sequence diagram of the Social Space communicator.

3) *Social Space client UI for PC*. This KP defines the default user interface of the developed social network client for PCs. It handles recognition of the user requests and actions and forwards corresponding commands to the service processing KPs. The sequence diagram of the corresponding implementation of PC UI KP is presented in Figure 12.

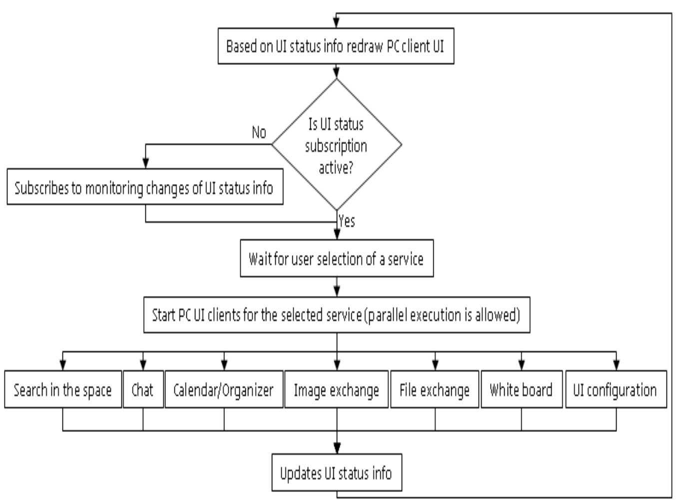


Figure 12. Sequence diagram of the Social Space Client UI for PCs.

Please note that the set of basic services defined in the figure above is just an initial basic set, which later based on users' feedback will be extended by the new services.

4) *Social Space client UI for mobile devices, e.g., for Symbian/MAEMO*. The KP defines the default user interface layout of the developed social network client for Symbian mobile devices. Similar KP for MAEMO platform is currently under development and the solution can be easily extended to any other mobile platforms, which requires implementing only more of the same type of KPs, which will better adopt UI restrictions of the other OS. The sequence diagram of the corresponding KP implementation for Symbian OS is presented in Figure 13.

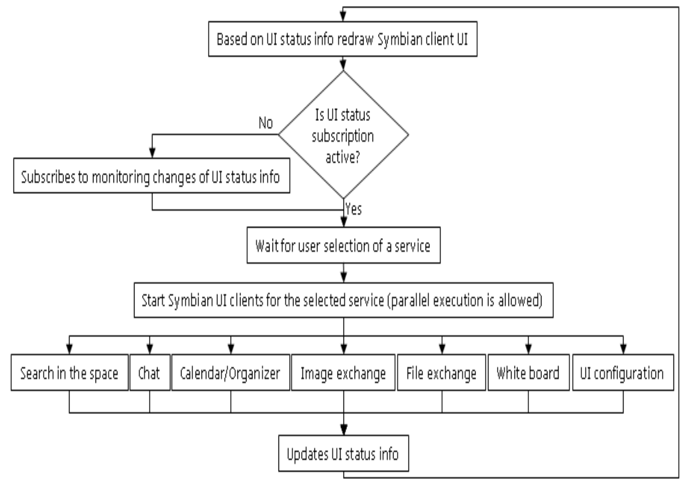


Figure 13. Sequence diagram of the Social Space Client UI for Symbian mobile devices.

5) *Gateway interface to the user accounts in other social networks, e.g., Facebook*. This class of KPs provides interface to user's accounts in other social networks, e.g., Facebook, and saves in SIB structure the information flows, which are coming through our social network client to the extend authorized by the user, the simplest example is by extracting some user-relevant information from analysis of chat messages. Later similar KPs for all popular social networks will be developed and integrated into the default distribution of the package. In Figure 14 you can find the sequence diagram of the corresponding KP gateway implementation for Facebook.

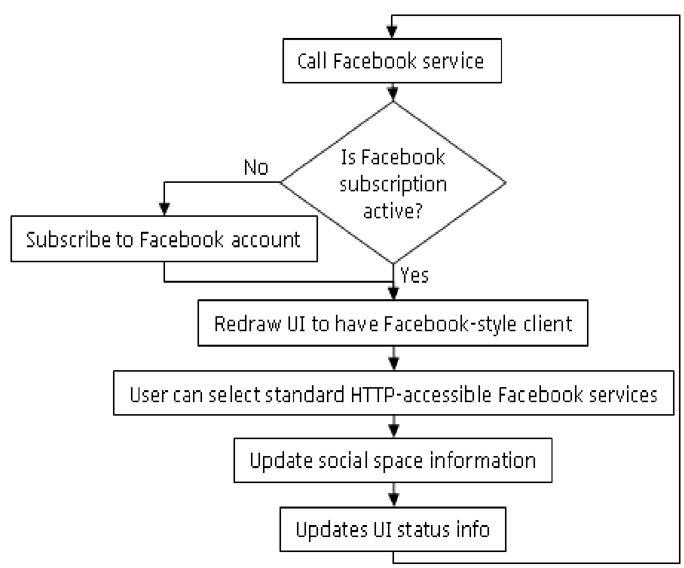


Figure 14. Sequence diagram of Facebook gateway.

6) *Gateway to the large service provision platforms/servers, e.g., OVI*. This class of KPs provide interface for easy and efficient access to the large external service repositories, e.g., Nokia OVI services repository [11]. Through this KP the user gets access to the huge variety of different services and so can use them to form the basic set of pre-provided services in the personal smart space. The similar solutions will be later created to interface other service provision platforms. The sequence diagram of the corresponding OVI gateway KP implementation is presented in Figure 15.

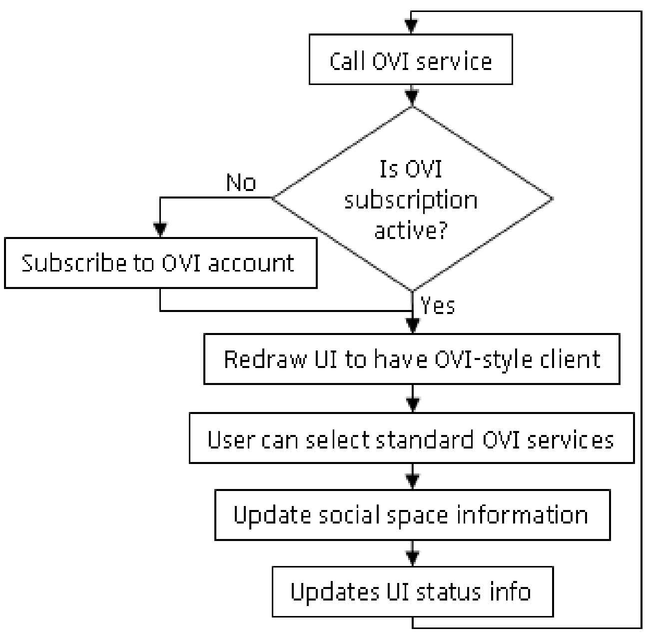


Figure 15. Sequence diagram of the OVI gateway.

The above listed set of KPs forms the very basic group of elements required to define the Smart-M3 social network client functionality. The first implementation of the corresponding application is currently in pilot testing phase. This project is run within scope of the Finnish-Russian University Cooperation in Telecommunication (FRUCT) program [12], which is nowadays one of the largest Open Innovation Academia-to-Industry communities in the Baltic region. The first publicly available solution is expected to be released by the next FRUCT seminar that will take place in the end of April 2010 in Saint-Petersburg, Russia. More information about the project progress can be found from the FRUCT website [12].

V. Discussion and Conclusions

This paper introduces the Smart-M3 platform that allows development of smart space solutions for seamless operation from PCs and mobile devices. The paper gives an overview of main issues that one should take into account in smart spaces design, defines main principles and provides reference model of our Smart-M3 solution, gives general idea about the provided developer's platform and in details discusses one application implemented on top of Smart-M3.

An interesting conclusion that one can derive from this study is about the role of mobile device in the future smart spaces. It is natural for mobile devices to become the personalized access point and interface to the surrounding Smart Spaces due to their availability to the users and their significant processing and storage capabilities. For example, the management functionality should inform the Smart Space about the user preferences and see how to obtain the favorite service of the user from the modules available in the given space. By having access to a large amount of personal information (e.g., calendar, email, etc.) and being carried by the user, the device can learn about the individual preferences and thus find or build up new services and offer them to the user at the most convenient time.

This is especially true when thinking about the social networking application area. This application area is very relevant and important, especially nowadays when more and more joint distributed project are initiated and there is clear understanding that industry and academia must collaborate closely. Creation of a social network for supporting distributed project work, with a feature of automatic data processing and reasoning would be of a great help. The closest nowadays available solution is Ning [13] online platform for creating own social networks. This solution is implemented on top classical web interface, but it is appealing to people who want to create their own social networks around specific interests with design, choice of features and member data. The Ning allows anyone to create own social network for a particular topic or need, catering to specific membership bases. Ning does allow developers to have source level control of their social networks, enabling them to change features and underlying logic. However, despite the significant step towards highly flexible and personalized social network, this solution has a number of critical limitations. It is implemented centralized principle of data storage and web access, which is not suitable for many commercial projects, as it is a key to keep full control over the data and preferably physically have the data only on the devices of member organizations. Secondly, Ning service does not provide good internal tools for data analysis and reasoning, so its integration into smart spaces solution is related to a number of problems. Finally, Ning service is PC platform oriented and porting it to mobile devices will face most of the problems mentioned in Introduction section.

In this paper we propose the new type of social networks solution that makes proactive analysis and reasoning of the going through information flow and derives new knowledge entities and relations for the personal smart spaces of the user. The solution provides a user with flexibility in selecting services and way of UI representation for each access device that the user might want to use. In addition a toolkit for easy development of new personalized modules is provided as a part of standard M3/Sedvice platform. Thanks to the great flexibility we can create equality efficient service clients for PC and mobile devices. Independently of the access device type, the service guaranties access to the same pool of data and preserving successive user experience. Also the proposed architecture maximizes application efficiency, decrease use of all resources, including energy and network traffic, and improves usability of UI. The key here is that the application does not need to contain redundant modules, which are not demanded by a given user. Also thanks to underlying Smart-M3 platform, the solution not only makes reasoning over collected data, but allows easy information exchange with other services. Another important feature for the users is that the solution allows use of the proposed social network together with other popular social networks, which only requires development of the corresponding knowledge processors. We believe that with the time, the social network solutions build on top of smart spaces platform will replace current web-based solutions and become dominant social network solution.

ACKNOWLEDGMENT

The authors would like to acknowledge input of the Sedvice/M3 group of Nokia Research Center, Helsinki and Smart-M3 FRUCT social network implementation team located at Saint-Petersburg State University of Airspace Instrumentation.

This study has been partially funded by TEKES ICT SHOK DIEM (www.diem.fi) project.

REFERENCES

[1] S. Balandin, I. Oliver, S. Boldyrev, “Distributed architecture of a professional social network on top of M3 smart space solution made in PCs and mobile devices friendly manner,” Procedings of 3rd Int. Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies (UBICOMM 2009). IEEE Computer Society, 2009, pp. 318–323.

[2] M. Williams. InfoWorld: More mobile Internet users than wired in Japan. http://www.infoworld.com/d/networking/more-mobile-internet-users-wired-in-japan-259. 2006.

[3] I. Oliver. Design and Validation of a Distributed Computation Environment for Mobile Devices. Proceeding of European Simulation Multiconference: Modelling and Simulation 2007. Malta. 2007.

[4] I. Oliver, J. Honkola. Personal Semantic Web Through A Space Based Computing Environment. Proceedings of Middleware for Semantic Web 08 at ICSC'08. Santa Clara, USA. 2008.

[5] A. Smirnov, N. Shilov, A. Krizhanovsky, A. Lappetelainen, I. Oliver, S. Boldyrev. Efficient distributed information management in smart spaces. ICDIM, pp. 483-488. 2008.

[6] D. Cook and S.K. Das. Smart environments: Technology, protocols and applications. John Wiley & Sons. ISBN: 0-471-54448-5, 2004.

[7] S.K. Das. Designing Smart Environments: Challenges, Solutions and Future Directions. Proceedings of ruSMART conference, St. Petersburg, Russia, 2008.

[8] I. Oliver, E. Nuutila, T. Seppo. Context gathering in meetings: Business processes meet the Agents and the Semantic Web. Proceedings of the 4th International Workshop on Technologies for Context-Aware Business Process Management, 2009.

[9] J. Jantunen, I. Oliver, S. Boldyrev, J. Honkola. Agent/Space-Based Computing and RF memory Tag Interaction. Proceedings of the 3rd International Workshop on RFID Technology - Concepts, Applications, Challenges, 2009.

[10] Sedvice/M3 group of NRC. Smart-M3 download page for developers. At SourceForge. http://sourceforge.net/projects/smart-m3. 2009.

[11] Nokia. The official webpage of OVI service platform. http://www.ovi.com/services/. 2009.

[12] A. Dudkov, S. Balandin. The official webpage of Finnish-Russian University Cooperation in Telecommunications (FRUCT) Open Innovation program. http://www.fruct.org. 2009.

[13] M. Andreessen, G. Bianchini. The official webpage of NING – online platform for creating own social network. http://www.ning.com. 2009.

Using Convolution Filters for Energy Efficient Routing Algorithm in Sensor Networks

Yaser Yousef
University of Haute-Alsace
34 rue du Grillenbreit
F-68000 COLMAR, France
yaser.yousef@uha.fr

Marc Gilg
University of Haute-Alsace
34 rue du Grillenbreit
F-68000 COLMAR, France
marc.gilg@uha.fr

Pascal Lorenz
University of Haute-Alsace
34 rue du Grillenbreit
F-68000 COLMAR, France
pascal.lorenz@uha.fr

Abstract

Several protocols and algorithms are used for energy efficient routing in Wireless Sensor Networks (WSNs). These protocols enable not only to reduce communication latency in these networks but also to maximize the network lifetime that is directly related to energy sensors due to the constraint of the batteries. Some of these protocols use clustering or avoiding holes to reach this goal. We have proposed, in this paper, a new hybrid algorithm based on clustering and convolution filters for efficient routing in terms of energy. For this reason, we present the network as a grayscale image. Our algorithm happens in two steps, each of these steps uses a different convolution filter. One kernel filter is used to determine the nodes that participate in the routing by using the mean filter. Another kernel is to pass packets from a source to the destination by using the gradient to select the routes over the most efficient battery.

Keywords - wireless sensor network, energy efficient routing, clustering, matrix convolution, kernel filter.

1. Introduction

During the last decade, wireless communications have emerged enabling users an access to information and electronic services regardless of their geographical positions. New applications of these technologies appear constantly. They have attracted great interest among individuals, businesses and industry.

One of these wireless communications is the Wireless sensor networks which are considered as a specific type of ad hoc network [1]. Indeed, the WSN share with MANET (Mobile Ad hoc Networks) several properties, such as the lack of infrastructure and wireless communications. But one of the key differences between the two architectures is the scope.

Wireless sensor networks are composed by a set of embedded processing units, called nodes, communicating via wireless links.

The goal of a WSN is to collect a set of environmental parameters surrounding nodes, such as temperature or pressure of the atmosphere in order to transport them to the point of treatment.

Unfortunately, the WSN are not a perfect solution, because of their low cost and their deployment in hostile areas, the sensors are quite fragile and vulnerable to various forms of failure: breakdown, non-rechargeable batteries... etc.

As sensor nodes are equipped with limited battery energy, it is also necessary to minimize their energy efficiency. The factor "energy" is at the center of the research around the sensor networks. This energy is mainly due to the air interface.

A sensor, often has limited energy and battery replacement is impossible. This means that the lifetime of a sensor depends greatly on the life of the battery. In a sensor network (multi-jumps) each node collects data and sends / transmits values. The failure of some nodes requires a change of network topology and the re-routing of packets. All these operations consume energy and therefore the current research focuses primarily on ways to reduce consumption.

The routing protocols for WSN have been widely studied, and various studies have been published. These protocols should be able to choose paths to route packets based on the actual condition of the network by increasing its lifetime and reducing the convergence time.

Clustering is one of the major approaches to treat the structure of a wireless sensor networks. It allows the formation of a virtual backbone that improves the use of limited resources such as bandwidth and energy. Furthermore, clustering helps to achieve multiplexing between different clusters. In addition, it improves the performance of routing algorithms. Several protocols use this preventive approach.

In a context of mobility and economy of node energy resources, self-configuring network is a problem in sensor networks. To preserve the battery of a node, it is necessary that its radio transmitters switch off as often as possible transition. Then, the problem of the synchronization of the nodes and the distribution waking periods appears. MAC layer is a solution that allows nodes to have sleep phases, without affecting communication. Several solutions at the MAC layer have been proposed to save energy. The role of MAC protocols is to organize the medium access between nodes wishing to communicate and to allow proper coordination between the nodes to share access to the medium but also to minimize node energy dissipation.

In this paper, we present an algorithm to improve and to complete our previous work in [2] and [3]. In this algorithm, we present the wireless sensor network as a grayscale image to achieve energy efficient routing. The pixels in this image represent the nodes in the WSN. We cut the image into sub-units or areas around a central node to determine the energy distribution in an area around this node. White regions in this image represent regions of the network that have sensors with a full battery. Unlike black regions which have nodes with empty batteries. We have used matrices and products of convolution in this algorithm using two filters. These two filters are used in the techniques of image processing; the first (the mean filter) is effective in identifying the nodes that will participate in the routing system. The second is to perform the routing of packets through the routes which have more energy efficiency (through the nodes that have the greatest capacity).

In section 2, we present a reminder of our previous work. Energy sector is explained in section 3. The Sections 4 and 5 describe our proposal. Results are presented in section 5. These are tested by the simulator OMNET++.

2. RELATED WORK

Several protocols have been proposed to conserve energy in WSNs. Scheduling protocols are one of these protocols, they are efficient in energy use because they avoid collisions and overhearing. They do not allow peer-to-peer communication and generally require cluster nodes. The inter-cluster communication is achieved by the two approaches: TDMA and FDMA [4].

Others, such as SMACS (Self-organizing Medium Access Control for Sensor Networks) proposed by Sohrabi and Pottie [5], which is a MAC protocol combining TDMA and FDMA in which the neighboring nodes randomly choose a slot and a frequency defining link.

Several solutions at the MAC layer have been proposed [4, 5, 6]. This layer allows nodes to have sleeping phases. For the MAC layer, two topologies are supported by the 802.15.4 protocol, the star topology and the peer to peer topology. In the star topology, communications are established directly between the central node (coordinator) and the sensors. The coordinator is the node that initiates and manages the network communications. The topology peer to peer allows each network node to communicate with any other node.

S-MAC (Sensor MAC) [7] is a similar protocol than 802.11. It uses the medium access CSMA / CA RTS / CTS (Request-To-Send, Clear-To-Send) that avoids collisions and hidden node problem [8]. The protocol establishes a mechanism for a standby distribution to each node in order to reduce energy consumption and extend its life. Each node should coordinate and exchange information with its neighbors to choose its own "sleep / active" cycle.

Other types of energy efficient routing protocol in WSNs have been created as the hierarchical protocols with clustering, cluster-head, aggregation and data fusion to reduce the total energy of the network by limiting communications between nodes. Consequently, the routing is done between clusters and not between nodes [9, 10, 11].

In [12, 13], protocols have been deployed to maximize the lifetime of the network by avoiding regions with less battery to choose the route that passes through the regions with high battery capacity.

As mentioned above, this paper is an improvement of our previous work in [1] and [2].

In [1] and [2], we used an image processing algorithm to get information about energy repartition in the network. To do this, we represented the network energy capacity by a grayscale image. The pixel with coordinates X, Y represents the sensors located at X, Y and the gray level is defined by the energy capacity C of this sensor. Any sensor with a full battery is represented by a white pixel, and the sensor with an empty battery is referred to by a black pixel. In this representation, the adaptive algorithm routes packets to the light part of the image to preserve the battery capacity.

An adaptive routing algorithm based on image processing algorithm is represented in [1]. It utilized the gradient to select an energy efficient path. In this algorithm, we obtained the energy gradient in an image by the convolution product computed by the Sobel algorithm which will be detailed in Section 4.

In [2], we used the convolution matrix and clustering to choose the more economical energy path to minimize energy consumption in the entire network.

3. NETWORK MODEL AND POWER MODEL

Before presenting our algorithm, we must mention that we work in a sensor network that has a limited capacity C. The sensors are often in isolated locations, and in some application domains, it is almost impossible to replace their batteries. So, we must conserve their capacities because the failure of a sensor could cutoff communications in the entire network. This capacity must be used effectively to extend the network lifetime. These sensors communicate with each other through wireless links in which packets go in one direction or in an alternative direction but not simultaneously. This type of connection provides a bidirectional link using the full capacity of the network. Each node is capable of receiving and transmitting packets to a particular node chosen by the administrator called CH (Cluster-Head) who will coordinate the nodes. These packets are transmitted as UDP messages.

Energy is consumed primarily by reception and transmission operations [2, 3], which is used the following energy model:

$$E(t+\Delta t) = E(t) - E_{tran} * \Delta t_{tran} - E_{recv} * \Delta t_{recv} - E_{idle} * \Delta t_{idle} - E_{sleep} * \Delta t_{sleep} \quad (2)$$

$$\Delta t = \Delta t_{tran} + \Delta t_{recv} + \Delta t_{Eidle} + \Delta t_{sleep} \quad (3)$$

E_{tran}, E_{recv} denote energy consumption for transmission and reception.

E_{idle}, idle energy, refers to the energy spent in idle mode.

E_{sleep} denotes energy expenditure in sleep mode.

Δt_{tran} , Δt_{recv} denote the time of transmission, reception energy.

Δt_{Eidle} denotes the time of energy spent in idle mode.

Δt_{sleep} denotes the time of energy spent in sleep mode.

4. ENERGY SECTORS AND ASSOCIATED MATRIX

After representing a wireless sensor network as a grayscale image, we can cut the neighborhood of a node into sectors in order to compare the energy of this node relative to its neighbors. We cut the region around a pixel (sensor) in eight or in twenty-four sectors as shown in Figures 1.

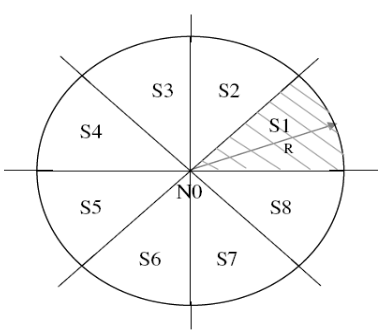


Figure 1. Energy sector

The goal of this representation is to know the distribution of energy in the network [14, 15]. We can see that the network as an image with white and black areas. Consequently, we represent the wireless sensor network like a map of pixels. Each pixel is identified by both an x and a y coordinate as shown in Figure2. In the context of digital images, we can use a grid of rows and columns in which is reserved a place to store the value of each pixel of the image. In mathematics this type of chart is called a matrix.

$P_{x-1,y-1}$ b	$P_{x,y-1}$ b	$P_{x+1,y-1}$ b
$P_{x-1,y}$ b	$P_{x,y}$ a	$P_{x+1,y}$ b
$P_{x-1,y+1}$ b	$P_{x,y+1}$ b	$P_{x+1,y+1}$ b

Figure 2. Pixel's image

In our case, the value related to the brightness of a pixel represents the energy capacity of a node. Hence, each sector will have the mean energy used to create the energy matrix M. This matrix will be used to calculate the product convolution [16].

$$M = \begin{pmatrix} s_3 & s_2 & s_1 \\ s_4 & s_0 & s_8 \\ s_5 & s_6 & s_7 \end{pmatrix}$$

5. MATRIX CONVOLUTION

A convolution is a mathematical operation which is used to multiply matrices together. In our case, we multiply two matrices:

The first is the image matrix that represents the energy values of nodes around a central node (energy matrix).

The second is a matrix called the kernel, noted as K. This matrix is the "heart" of all the changes that will affect the image. The kernel will act on each pixel, i.e. on each element of the matrix image [17].

The kernel is composed by the square matrix [k] of 3x3 elements. We apply a convolution filter to multiply each pixel of the image matrix M by the kernel [k]. To calculate the value of a pixel (x, y) of the image matrix, we multiply its value by the pixel central kernel K (2.2) shown in the Figure 3 and summing up the product value of adjacent pixels.

K (1,1)	K (2,1)	K (3,1)
K (2,1)	K (2,2)	K (2,3)
K (3,1)	K (3,2)	K (3,3)

Figure 3. Kernel matrix

Then the convolution product of two n+1×p+1 matrix K and M is defined by the formula:

$$K * M = \sum_{i=0}^{n} \sum_{j=0}^{p} K_{i,j} \times M_{n-i,p-j} \qquad (1)$$

To perform the convolution product of two functions we take an average of these two functions. It is widely used in mathematics to approximate and stabilize functions.

6. KERNEL MATRIX

This work will proceed in two phases and each phase uses a different filter to find the most economical energy path from the source to the destination. The two kernels used are:

6.1 1D : 1 kernel matrix K (mean filter)

As shown in Figure1, we have eight energy sectors. In our proposal, we will use the following kernel matrix:

$$K = \begin{pmatrix} 1/12 & 1/12 & 1/12 \\ 1/12 & 4/12 & 1/12 \\ 1/12 & 1/12 & 1/12 \end{pmatrix}$$

In this way, we will use the mean filter that is simple, intuitive and easy to implement for smoothing images. This filter represents the shape and size of the neighborhood to be sampled when calculating the mean. It replaces each pixel by the average values of adjacent pixels and the central pixel. Therefore, it smoothes the signal and the filter works as a low-pass one. This filter enables any sensor of the network to take an average across its neighborhood. In this configuration, the central node will contribute for half weight compared to the eight surrounding regions. Doing this, a high energy capacity central node will be chosen to forward the packets, even if it is located in a poor energy region.

6.2 2D: 2 kernel matrices Kx,Ky (Sobel)

In our routing, after selecting the active nodes, we have to select the most economical energy routes. In this step we use the following matrices:

$$K_x = \begin{pmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{pmatrix} \qquad K_y = \begin{pmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix}$$

These matrices are used by the algorithm of Sobel [18] to obtain the gradient in grayscale image. The Sobel filter uses two 3x3 kernels, one for the horizontal axis (X) and the other for the vertical (Y). Each core is a gradient filter, which is them combined to create the final image.

The energy gradient $G_{i,j} = (G_x, G_y)$ at sensor i, j used by Sobel, it is obtained by the following equations:

$$G_x = M_{i,j} * K_x \qquad (4)$$

$$G_y = M_{i,j} * K_y \qquad (5)$$

Where $M_{i,j}$ is a 3 × 3 sub matrix of the grayscale image that represents the sensors energy capacities.

In the Sobel algorithm $G_{i,j}$ represents the gradient of gray intensity at pixel i, j. This intensity is related to the battery capacity of sensor i, j by definition.

The norm of the gradient is given by:

$$\|G\| = \sqrt{G_x^2 + G_y^2} \qquad (6)$$

and the direction of the gradient is given by:

$$\theta = \arctan\left(\frac{G_y}{G_x}\right) \qquad (7)$$

The product of multiplying matrix M by matrices K_x, K_y on the two axes x and y is then:

- On the x-axis shown in Figure.4:

$G_x = M_{i,j} * K_x$

$$= \begin{pmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{pmatrix} * \begin{pmatrix} s_3 & s_2 & s_1 \\ s_4 & s_0 & s_8 \\ s_5 & s_6 & s_7 \end{pmatrix}$$

$$= S_7 + 2S_8 + S_1 - S_3 - 2S_4 - S_5 \qquad (8)$$

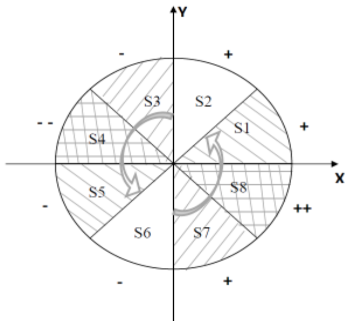


Figure 4. Gradient direction x

We can notice that the direction of the gradient on the x-axis is influenced by six values including three positive values (S_7, $2S_8$, S_1) and three negative ($-S_3$, $-2S_4$, $-S_5$). The central energy sector on each side (S_8 and S_4) is the double of the energy of its neighboring areas.

- On the y-axis shown in Figure.5:

$G_y = M_{i,j} * K_y$

$$= \begin{pmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix} * \begin{pmatrix} s_3 & s_2 & s_1 \\ s_4 & s_0 & s_8 \\ s_5 & s_6 & s_7 \end{pmatrix}$$

$$= S_5+2S_6+S_7-S_1-2S_2-S_3 \qquad (9)$$

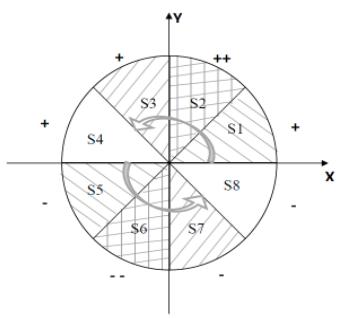


Figure 5. Gradient direction y

We can notice that the direction of the gradient on the y-axis is influenced by six values including three positive values (S_5, $2S_6$, S_7) and three negative ($-S_1$, $-2S_2$, $-S_3$). And the central energy sector to each side (S6 and S2) is the double of the energy of its surrounding areas.

These two kernels will be used in the algorithm that we will explain in the next section.

7. HYBRID ALGORITHM

The failure of a sensor can have serious consequences on the total life of the network, so it is necessary to involve only the nodes that have a capacity greater than a certain threshold (called ET). The sensors that do not participate in routing will be dormant for the period of pending instructions from HC to wake up and participate in routing.

7.1 Parameters of our algorithm

- X_i, Y_i: positions of sensors N_i.
- CH: Cluster-Head.
- BC: Battery capacities.
- ET: Energy Threshold, if sensor's battery is less than this value, the forwarding packets over UDP will not happen.
- SEID: Sending Energy Information Delay, the delay between two information packets.
- T: Time of energy packet sent (current time – last time).
- $T_{sleep:}$ sleep time.
- R: Range, for energy sectors.
- K: Kernel matrix.
- M: Energy matrix of N_i.
- ERP: Energy Routing Parameter: K*M, the average energy around the node.

This work will proceed in three phases in order to find the most economical energy path from source to destination.

7.2 Energy Routing Algorithm steps

7.2.1 Exchange energy information and convolution computations

1. Sending energy information:
   If (BC_i change) then
   If (T > SEID) and node is sleeping
   Then send (BC_i and X_i, Y_i) over UDP to CH.
2. Convolution computations (running on CH):
   a) Obtain energy information from Ni.
   b) update energy table :
      - IP address of N_i
      - X_i, Y_i
      - BC_i
   c) Compute:
      - Compute energy matrix: M of N_i
      - ERP_i =K*Ms_i
      - G_{xi} , G_{yi}
   d) Send ERP to all sensors over UDP.

7.2.2 Waking state and sleeping state

3. On each sensor: routing packet and IP routing process*:*
   For each sensor, get ERP, G_{xi} ,G_{yi} over UDP.
   If (ERP < ET)
   Then sleep during t_{sleep}
   And if (sent packet is not a control packet)

Then sensor is set in standby mode

4. Sensor wake-up during time t_{sleep} then it request a new K*M value

If (k*M> ET) then end standby mode

7.2.3 Routing packets

5. Compute vector $V=(K_x*M-S_x, K_y*M-S_y)$ at sensor (S_x,S_y)
6. Compute cos α=cos (V,G) where G is next-hop vector and

$$COS\left(V_{x',y'},G_{x,y}\right)=\frac{\vec{V}_{x',y'}\times\vec{G}_{x,y}}{\left\|\vec{V}_{x',y'}\right\|\times\left\|\vec{G}_{x,y}\right\|} \quad (10)$$

7. Compute the minimums values of cos α.
8. Select the path with the maximal of the minimums values of cos α.

The two following charts represent our algorithm:

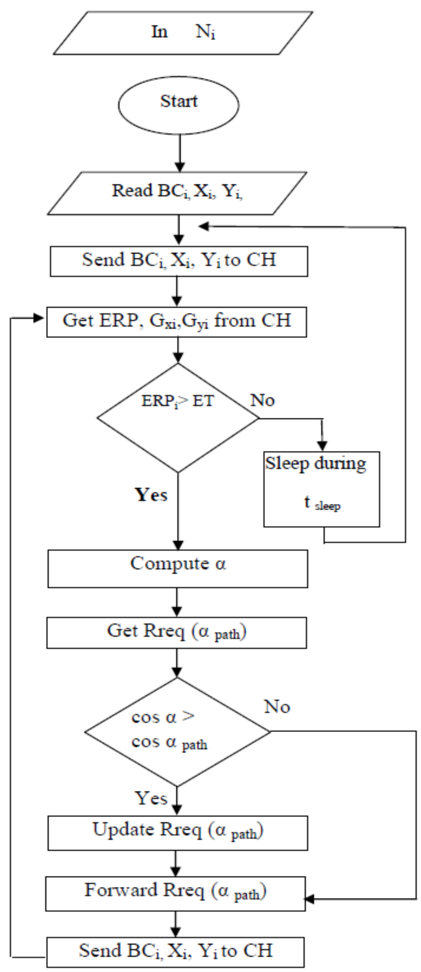


Figure 6. Running on N_i

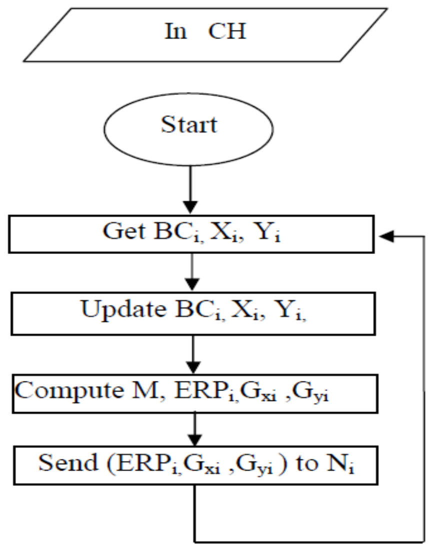


Figure 7. Running on CH

Therefore, we can resume the previous three phases of this algorithm as:

1D is used to obtain the energy information of each node by the C-Hs.

2D is used to obtain the direction of the gradient indicating the searched route through which passes the packets.

8. SIMULATION

Our work is done in a sensor network which has the topology shown in figure 8 where N1 is the source, N3, N4 are the gateways and N0 is the destination.

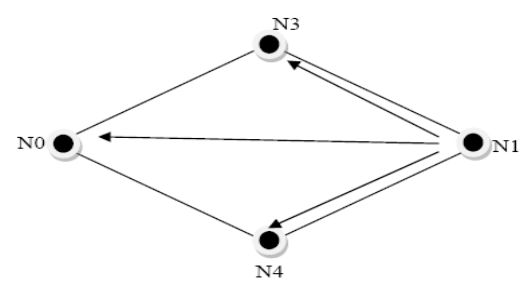


Figure 8. Network topology

Results shown in the tables 1, 2, 3, 4, 5 and 6 represent the average of twenty simulation runs performed by the simulator OMNET++ that had been modified by implementing the two kernels (Sobel and mean filter) [18, 19]. We have used the device Nano WiReach as a model that acts as a bridge to connect serial devices to 802.11b/g Wireless LANs [20]. Its power Consumption: 250mA in transmission, 190 mA in reception (typical), 8 mA in sleep and 8 mA for idle.

We have used the AODV routing protocol. Control packets that have a size of 512 Bytes are sent using UDP protocol. Node N1 sends packets as a burst of a frequency of 0.01s with a sleep delay of: 0, 50, 100 and 150s. Initial values of energy given to N0, N1, N3, and N4 in our simulations are respectively: 40mA, 20mA, 2mA, and 10mA, where N0 is the CH in our configuration. The energy threshold has the values: 0mA, 1mA, 2mA, 3mA, 4mA and 5 mA. Total simulation time is 1000 seconds.

The following three tables 1, 2 and 3 indicate the simulation results for the number of sent packets.

Sleep delay	Threshold (0)				Threshold (1)			
	Host0	Host1	Host3	Host4	Host0	Host1	Host3	Host4
0	$2.41e^{+4}$	$1.06e^{+5}$	$3.32e^{+4}$	$9.72e^{+4}$	$2.45e^{+4}$	$1.06e^{+5}$	$2.54e^{+4}$	$9.33e^{+4}$
50	$1.92e^{+4}$	$7.09e^{+4}$	$3.63e^{+4}$	$6.21e^{+4}$	$1.52e^{+4}$	$6.66e^{+4}$	$1.99e^{+4}$	$5.81e^{+4}$
100	$1.93e^{+4}$	$7.95e^{+4}$	$3.88e^{+4}$	$6.00e^{+4}$	$1.88e^{+4}$	$8.53e^{+4}$	$1.72e^{+4}$	$8.48e^{+4}$
150	$1.91e^{+4}$	$7.87e^{+4}$	$3.63e^{+4}$	$6.17e^{+4}$	$2.13e^{+4}$	$1.06e^{+5}$	$1.80e^{+4}$	$1.07e^{+5}$

Table 1: Packet sent number with ET of 0 and 1

Sleep delay	Threshold (2)				Threshold (3)			
	Host0	Host1	Host3	Host4	Host0	Host1	Host3	Host4
0	$2.12e^{+4}$	$8.50e^{+4}$	$1.22e^{+4}$	$6.02e^{+4}$	$1.82e^{+4}$	$6.75e^{+4}$	$1.25e^{+4}$	$2.36e^{+4}$
50	$2.13e^{+4}$	$1.06e^{+5}$	$2.82e^{+3}$	$1.19e^{+5}$	$7.21e^{+3}$	$5.11e^{+4}$	$1.25e^{+3}$	$3.74e^{+4}$
100	$2.08e^{+4}$	$1.06e^{+5}$	$1.88e^{+3}$	$1.21e^{+5}$	$6.85e^{+3}$	$4.53e^{+4}$	603.000	$4.01e^{+4}$
150	$1.88e^{+4}$	$9.85e^{+4}$	$1.03e^{+3}$	$1.13e^{+5}$	$6.65e^{+3}$	$4.25e^{+4}$	366.000	$4.00e^{+4}$

Table 2: Packet sent number with ET of 2 and 3

Sleep delay	Threshold (4)				Threshold (5)			
	Host0	Host1	Host3	Host4	Host0	Host1	Host3	Host4
0	$2.21e^{+4}$	$8.72e^{+4}$	$1.35e^{+4}$	$1.68e^{+4}$	$2.56e^{+4}$	$1.06e^{+5}$	$1.41e^{+4}$	$2.13e^{+4}$
50	325.000	$2.49e^{+4}$	306.000	41.000	321.000	$2.49e^{+4}$	305.000	42.000
100	110.000	$1.29e^{+4}$	23.000	98.000	110.000	$1.30e^{+4}$	24.000	97.000
150	92.500	$1.03e^{+4}$	83.000	19.000	92.500	$1.03e^{+4}$	82.000	18.000

Table 3: Packet sent number with ET of 4 and 5

In the three tables above, we see that there are a few packets sent by the node N0 as it is the destination. In Table 3, at a threshold of 4 and 5, we note that the number of packets that take the route between the source and the destination by both gateways (N3 and N4) tends to zero when the threshold is greater or equal to 4. In other words, the communication will be interrupted. We will focus on routing with a threshold below 4. At threshold between 2 and 3, energy has been saved because the number of packets sent by N4 decreases as shown in figure 9 and 10.

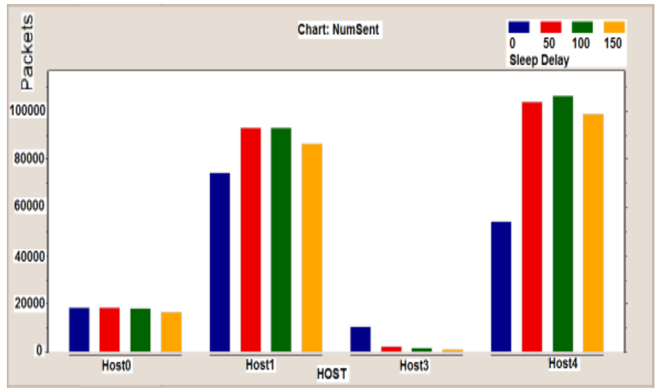


Figure 9. Packet sent number with ET of 2

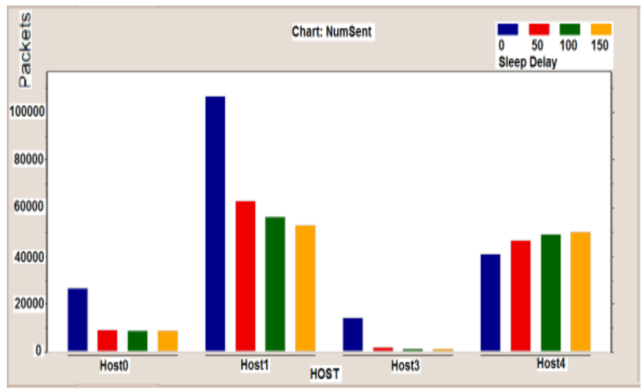


Figure 10. Packet sent number with ET of 3

The following tables represent the percentage of remaining energy in the network nodes; we note that the energy value remaining in N3 is zero because its initial value is proximate to the threshold and total simulation time is 1000s. Consequently, we will concentrate on the other gateway (node N4).

Sleep delay	Threshold (0)				Threshold (1)			
	Host0	Host1	Host3	Host4	Host0	Host1	Host3	Host4
0	0.909	0.826	0	0.603	0.911	0.830	0	0.609
50	0.870	0.694	0	0.307	0.884	0.666	0	0.301
100	0.867	0.736	0	0.358	0.868	0.799	0	0.443
150	0.866	0.734	0	0.353	0.912	0.830	0	0.609

Table 4: Energy Left with ET of 0 and 1

Sleep delay	Threshold (2)				Threshold (3)			
	Host0	Host1	Host3	Host4	Host0	Host1	Host3	Host4
0	0.879	0.761	0	0.426	0.839	0.771	0	0.342
50	0.912	0.831	0	0.608	0.933	0.865	0	0.724
100	0.912	0.831	0	0.608	0.936	0.867	0	0.722
150	0.907	0.800	0	0.550	0.936	0.868	0	0.723

Table 5: Energy Left with ET of 2 and 3

Sleep delay	Threshold (4)				Threshold (5)			
	Host0	Host1	Host3	Host4	Host0	Host1	Host3	Host4
0	0.890	0.771	0	0.503	0.922	0.839	0	0.698
50	0.944	0.881	0	0.779	0.944	0.811	0	0.779
100	0.948	0.885	0	0.778	0.948	0.885	0	0.778
150	0.949	0.886	0	0.778	0.949	0.886	0	0.778

Table 6: Energy Left with ET of 4 and 5

In tables 4 and 5, we see that with a T_{sleep} equal to 0 and a threshold between 0 and 3, the node N4 loses its energy. On the other hand, it retains its energy if the threshold is greater or equal to 4 as shown in tables 3 and 6, because it does not send packets (except control packets).

At a T_{sleep} of 50s and at a threshold of 1, node N4 has used 70% of its energy. At a threshold of 2 it has only used 40% and at a threshold of 3 or 4 or 5 it has used 24% of its capacity. That means, we have saved 30% of energy in combination (T_{sleep} = 50, ET = 2) as shown in Figure 11 and 12.

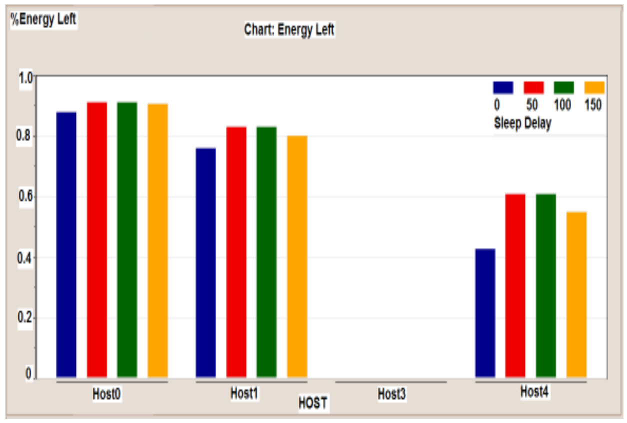


Figure 11. Energy left with ET of 2

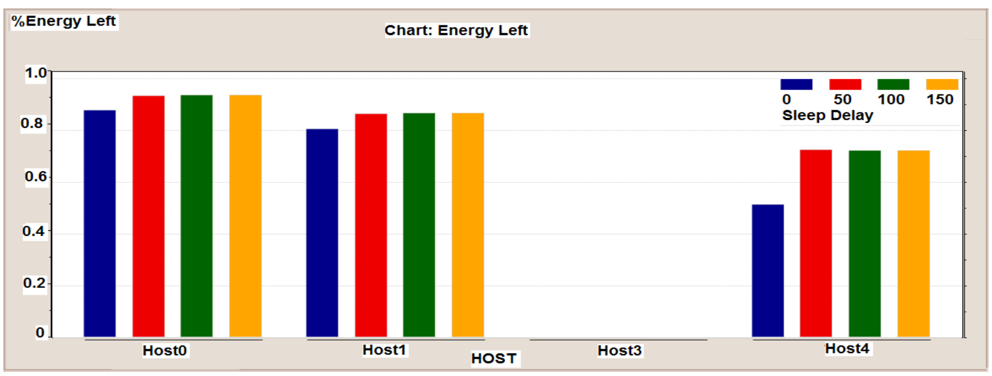


Figure 12. Energy left with ET of 3

At a T_{sleep} of 100s and at a threshold of 1, node N4 has used 55% of its energy. At a threshold of 2 it has only used 40% and at a threshold of 3 or 4 or 5 it has used 24% of its capacity. That means, we have saved 15% of energy in combination (T_{sleep} = 100, ET = 2).

If we increase the T_{sleep} to 150s. At a threshold of 0, node N4 has used 64% of its energy. At a threshold of 2 it has only used 40% and at a threshold of 3 or 4 or 5 it has used 24% of its capacity. But the number of packets forwarded by N4 decreases dramatically. If we continue to increase the threshold, the node stops to relay packets as shown in figure 13.

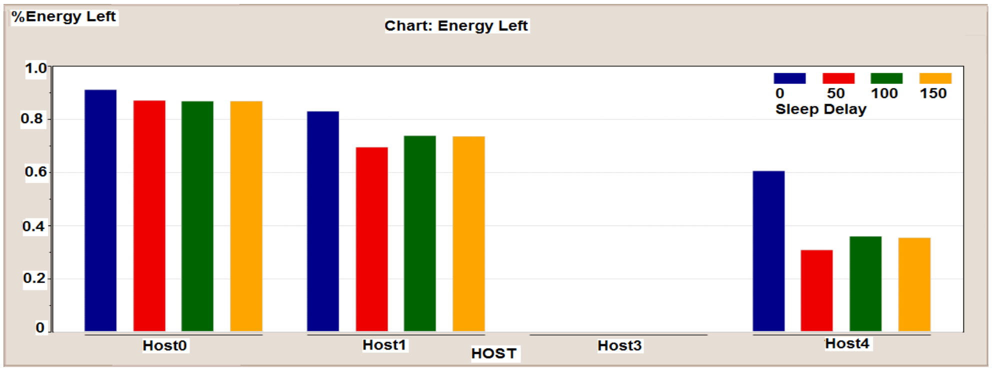


Figure 13. Energy left with ET of 0

Concerning energy consumption over time, it is clear that if we increase the sleep time, we save more energy as shown in Figures 14 and 15, which show the energy change of node N3 during the simulation time of 1000s.

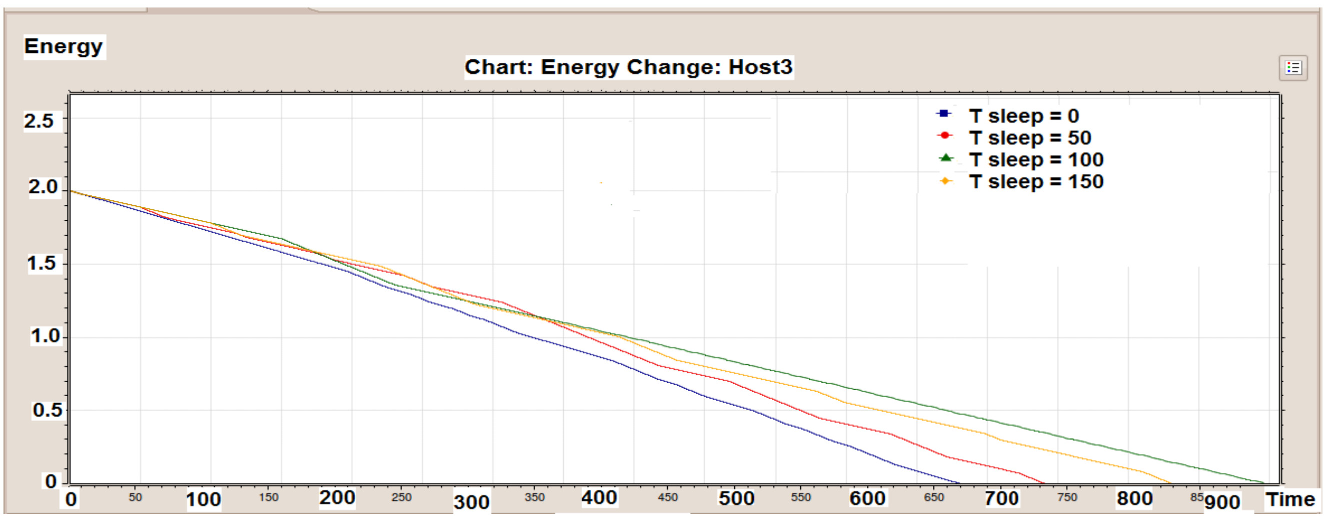


Figure 14. Energy Consumption of N3 with ET of 2

At a threshold of 2mA and a simulation time of 1000s, we remark that the node N3 stops transmitting and it is exhausted;

BC_i= 0 after 672 seconds when T_{sleep} = 0
BC_i= 0 after 739 seconds when T_{sleep} = 50
BC_i= 0 after 882 seconds when T_{sleep} = 100
BC_i= 0 after 823 seconds when T_{sleep} = 150

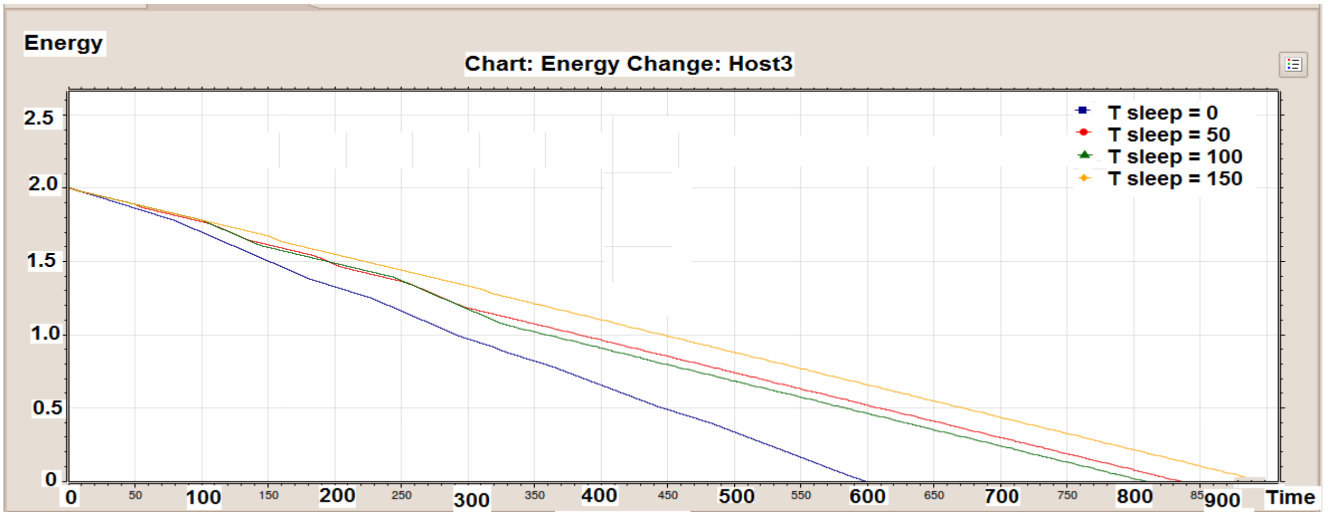


Figure 15. Energy Consumption of N3 with ET of 3

On the other side, at a threshold of 3mA;

BC_i= 0 after 598 seconds when T_{sleep} = 0
BC_i= 0 after 836 seconds when T_{sleep} = 50
BC_i= 0 after 810 seconds when T_{sleep} = 100
BC_i= 0 after 897 seconds when T_{sleep} = 150

Looking at the slope of the curve in Figures 14 and 15, we note that, the greater the sleep delay, the more the slope of curve is low (the less the node consumes its energy).

Our main objective is to extend lifetime. In the following graphs, which show the energy change of node N4 during the simulation time of 1000 seconds, we note that if we increase the sleep time to 150 and if we increase the energy threshold from 0 to 3, we achieve the best results as is shown in the following figures:

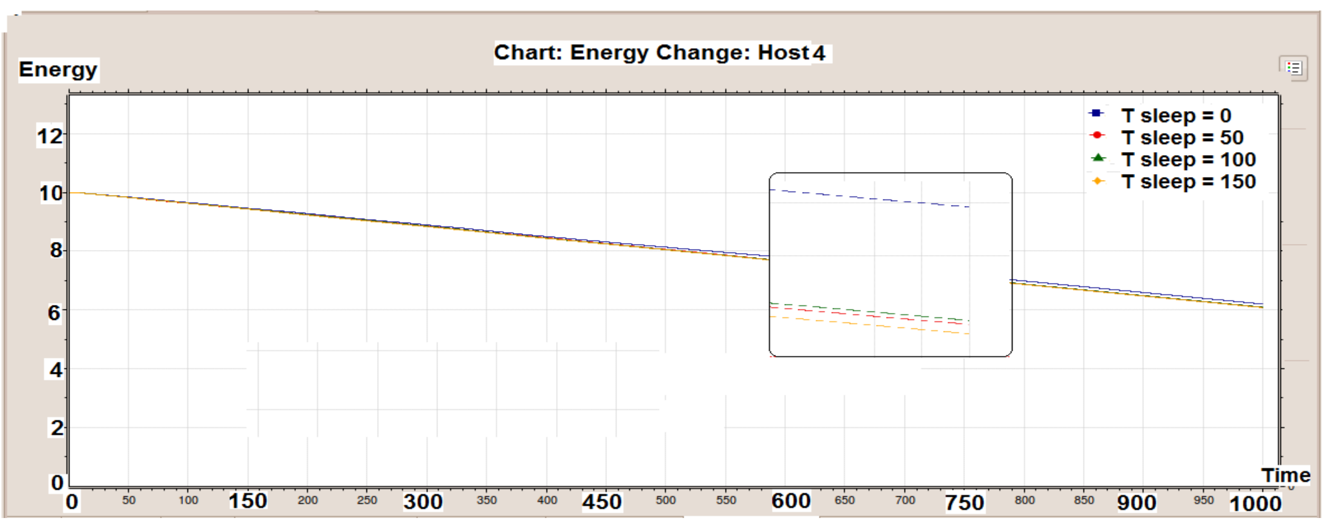


Figure 16. Energy Consumption of N4 with ET of 2

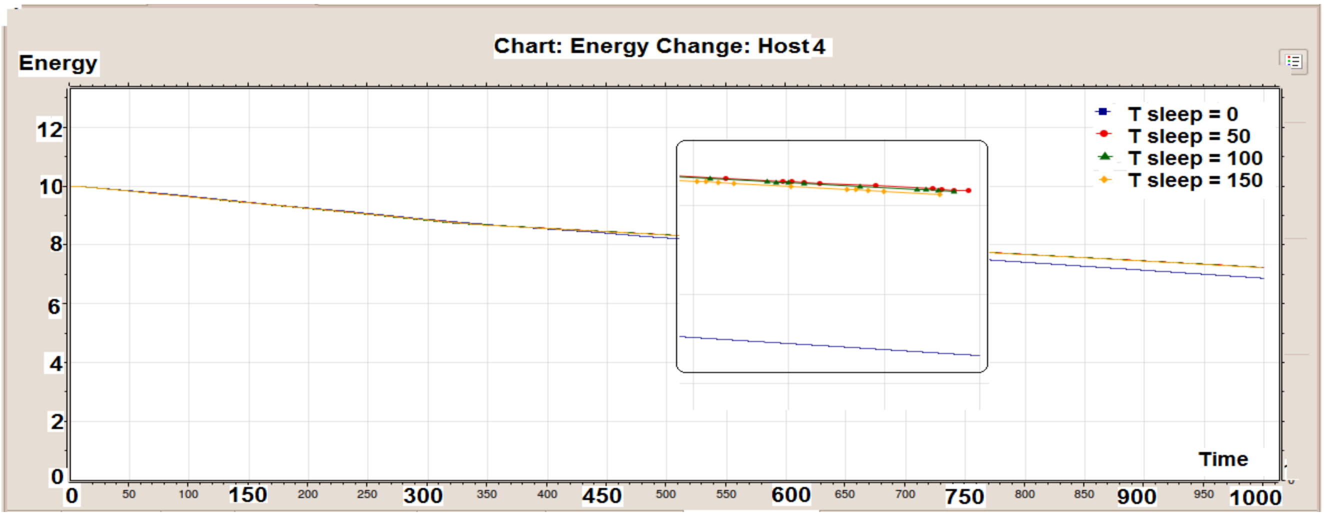


Figure 17. Energy Consumption of N4 with ET of 3

Consequently, we found that the best results are obtained with a T $_{sleep}$ of 50 and a threshold between 1 and 2, because we have saved 30% of energy.

Moreover, these results show the good performance of our algorithm because we have extended the lifetime of the node N4.

9. Conclusion and future work

In this paper, a new idea was created by treating the wireless sensor network as an image. This image consists of regions containing nodes. Our algorithm cuts the region around the central node into eight sectors. Each sector will have a value of energy used to create an energy matrix (3 × 3). We can use this matrix in the convolution. This parameter is used in the proposed routing algorithm to choose the path that passes through the nodes of a high capacity battery.

This algorithm is based on energy efficient routing. It also uses clustering in which all calculations are done by the CH which has CPU power, memory resources and energy capacity more important than other nodes. In addition, nodes that do not participate in routing are instantly dormant in order to conserve energy. They wake up after a fixed period to either participate in routing or remain dormant.

We have developed some components of the network simulator OMNET + + to adapt our algorithm

We have successfully saved the energy of nodes, i.e. we have retained 30% of the energy of N4. Consequently, we have maximized the total network lifetime.

In future work, other techniques of image processing, such as edge detection, to avoid the holes (empty regions where there are no sensors or there's not enough of energy). We can also use other filters to reduce the number of control messages.

References

[1] M. Gilg, A. Makhlouf,and P. Lorenz, “Fairness index in single and double star Network”. International Journal On Advances in sytems and Measurements”. Vol 2 no1 , 2009, pp. 109-108.

[2] M. Gilg, Y. Yousef, and P. Lorenz, “Using image processing algorithms for energy efficient routing algorithm in sensor networks”. ADAPTIVE 2009. Athens/Glyfada, Greece, November 2009. pp. 132-136.

[3] M. Gilg, Y. Yousef, and P. Lorenz, “Using Matrix convolutions and Clustering for Energy Efficient Routing Algorithm in Sensor Networks”. AICT 2010. Barcelona, Spain, may 2009.

[4] R. Lin, Z. Wang , and Y. Sun, “Energy efficient medium access control protocols for wireless sensor networks and its state-of-art ” .in Industrial Electronics, 2004 IEEE International Symposium, May 2004.pp 669 – 674.

[5] K. Sohrabi and G.J. Pottie “Performance of a novel self-organization protocol for wireless ad-hoc sensor networks”. In Proceeding of the 50th IEEE Vehicular Technology Conference (VTC’99), vol.2, Amsterdam, September 1999. pp.1222-1226.

[6] M. Caccamo, L. Y. Zhang, L. Sha, et G. Buttazzo. “An implicit prioritized access protocol for wireless sensor networks“. IEEE Real-Time System Symposium (RTSS’02), décembre 2002.

[7] W. Ye, J. Heidemann, and D. Estrin. “An Energy-Efficient MAC Protocol for Wireless Sensor Networks”. In Proceedings of the 21st International Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM’02), vol.3, New York, NY, USA, June 2002. pp.1567-1576.

[8] A. Rowe, R. Mangharam, and R. Rajkumar “RT-Link, A Global Time- Synchronized Link Protocol for Sensor Networks”. Elsevier Ad hoc Networks, Special Issue on Energy Efficient Design in Wireless Ad Hoc and Sensor Networks, 2007.

[9] N. V. Subramanian, “Survey on Energy-Aware Routing and Routing Protocols for Sensor

Networks". Technical Report, Computer Science, University of North Carolina, Charlotte, 2004.

[10] K. Akkaya, and M. Younis, "A Survey on Routing Protocols for Wireless Sensor Networks". Journal of Ad Hoc Networks, Vol. 3, No. 3, May 2005. pp. 325-349.

[11] Al-Karaki, J.N. Kamal, A.E. " Routing techniques in wireless sensor networks: a survey. Wireless Communications", IEEE, December. 2004.

[12] D. Kim, J. J. Garcia-Luna-Aceves, K. Obraczka, J. C. Cano, and P. Manzoni, "Routing Mechanisms for Mobile Ad Hoc Networks Based on the Energy Drain Rate". In IEEE Trans. Mob. Comput, 2003. pp. 161-173.

[13] Y. Tian, F. Yu, and Y. Choi, S. Park, E. Lee, M. Jin, and S. H. Kim, "Energy efficient data dissemination protocol for detouring routing holes in wireless sensor networks". ICC 2008. IEEE International Conference on Communications, vol. 31, no. 1, May 2008. pp. 2322-2326.

[14] M. I. Razzak, S. A. Hussain, A .A. Minhas and M. Sher, "Collaborative Image Compression in Wireless Sensor Networks". International journal of computational cognition (http://www.ijcc.us), vol. 8, no. 1, march 2010.

[15] L. Zhang , Y. Lu, L. Chen, D. Dong, "Game Theoretical Algorithm for Coverage Optimization in Wireless Sensor Networks". Proceedings of the World Congress on Engineering 2008 Vol I WCE 2008, London, U.K, July 2 - 4, 2008.

[16] B. Fisher, Y. Chen, "Generalizations of Non-Commutative Neutrix Convolution Products of Functions". Georgian Math, No. 5, 1997.pp. 413-419.

[17] R.G. Lanckriet, N. Cristianini,P. Bartlett,L. El Ghaoui and M. Jordan, "Learning the Kernel Matrix with Semidefinite Programming". JMLR.org, Volume 5,December 2004, pp27 - 72 .

[18] W. Drytkiewicz, S. Sroka, V. Handziski, A. Koepke, and H. Karl, "A mobility framework for OMNeT++," in Proceedings of the 3rd International OMNeT++ Workshop, Budapest, Hungary, January 2003.

[19] J. Glaser, D. Weber, S. A. Madani, and S. Mahlknecht, "Power Aware Simulation Framework for Wireless Sensor Networks and Nodes". In EURASIP Journal on Embedded Systems, Article ID 369178, May 2008.

[20] Nano WiReach™ "Miniature embedded secure WiFi module". In Connect One Ltd. Publication Number 17-3520-11, September 2008.

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
issn: 1942-2679

International Journal On Advances in Internet Technology
ICDS, ICIW, CTRQ, UBICOMM, ICSNC, AFIN, INTERNET, AP2PS, EMERGING
issn: 1942-2652

International Journal On Advances in Life Sciences
eTELEMED, eKNOW, eL&mL, BIODIV, BIOENVIRONMENT, BIOGREEN, BIOSYSCOM, BIOINFO, BIOTECHNO
issn: 1942-2660

International Journal On Advances in Networks and Services
ICN, ICNS, ICIW, ICWMC, SENSORCOMM, MESH, CENTRIC, MMEDIA, SERVICE COMPUTATION
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
issn: 1942-2628

International Journal On Advances in Systems and Measurements
ICQNM, ICONS, ICIMP, SENSORCOMM, CENICS, VALID, SIMUL
issn: 1942-261x

International Journal On Advances in Telecommunications
AICT, ICDT, ICWMC, ICSNC, CTRQ, SPACOMM, MMEDIA
issn: 1942-2601