Report from Dagstuhl Seminar 11181

Organic Computing – Design of Self-Organizing Systems

Edited by

Kirstie Bellman[1], Andreas Herkersdorf[2], and Michael G. Hinchey[3]

1 **Aerospace Corp. – Los Angeles, US,** `Kirstie.L.Bellman@aero.org`
2 **TU München, DE,** `herkersdorf@tum.de`
3 **University of Limerick, IE,** `mike.hinchey@lero.ie`

Abstract

This report documents the program and the outcomes of Dagstuhl Seminar 11181 "Organic Computing – Design of Self-Organizing Systems".

Seminar 1.–6. May, 2011 – www.dagstuhl.de/11181
1998 ACM Subject Classification F.1.1 Models of computation.
Keywords and phrases Organic Computing, Self-Organization, Adaptivity, Trustworthy System Design
Digital Object Identifier 10.4230/DagRep.1.5.1
Edited in cooperation with Johannes Zeppenfeld

1 Executive Summary

Kirstie Bellman
Andreas Herkersdorf
Mike Hinchey
Christian Müller-Schloer
Hartmut Schmeck
Rolf Würtz

Organic Computing (OC) has become a challenging vision for the design of future information processing systems: As the systems become increasingly more complex, powerful, cheaper and smaller, our environment will be filled with collections of autonomous systems. Autonomous systems are equipped with intelligent sensors and actuators to be aware of their environment, to communicate, and to organize themselves in order to perform the required actions and services. However, these abilities result in even greater system complexity, which we will not be able to explicitly design and manage in every detail, nor are we able to anticipate every possible configuration. Nevertheless, these services have to be as robust, safe, flexible, and trustworthy as possible. In particular, a strong orientation of these systems towards human needs – as opposed to a pure implementation of the technologically possible – is absolutely central.

So far, the OC community, mainly driven by the priority research program of the German Research Foundation (DFG), successfully proposed and – at least partially – established a common nomenclature and terminology for terms like emergence, self-organization, selfadaptation, robustness and flexibility within an interdisciplinary research community.

Quantitative metrics for emergence and self-organization were introduced and applied. Observer controller structures have been established as a common architectural pattern for OC systems within a wide spectrum of applications ranging from traffic control, to Systems on Chip, to collaborative robot systems, to wireless sensor networks. Roles and applicability of different types of supervised and reinforcement-based technical learning techniques were investigated and adapted to OC needs.

Despite the progress in understanding the implications and exploiting the potentials of the OC paradigm, a number of key challenges and research questions still remain. In particular, the planned 2011 OC seminar shall shed light on the various notions of design within the OC context. Design in the classical sense follows a hierarchical top-down constraint propagation starting from a purely functional specification. All eventual environmental influences and disturbances have to be anticipated by the designer at "design time". Due to this anticipatory nature the resulting system is rigid and not able to sufficiently react to run time events.

Complex systems in nature often develop bottom-up due to the self-organizing capabilities of their components. Each component and the system as a whole react to the demands of the environment. In doing so, they are guided by the principles to survive as an individual (selfishness) and the necessity to co-operate (altruism). In technical life-like OC systems we must provide some control by a higher-level entity (finally the user) guiding the bottom-up decisions of the components into a globally desirable direction.

In this way, the former top-down design process dissolves into a balanced run-time negotiation between top-down constraints and bottom-up opportunities. The ultimate consequence of this would mean a total replacement of the design process (at design-time) to controlled self-organization (at runtime).

The 2011 OC seminar was held to answer questions resulting from this shift from design-time to run-time. Is OC a realistic or even desirable vision? How can we replace rigid human designtime control by self-adaptive run-time control without stifling the creativity of the emergent bottom up processes? How can we balance top-down control and bottom-up emergence? Beyond these theoretical questions it is a goal of the seminar to define a number of concrete OC demonstrators – or even a common demonstrator – to be pursued in the sequel.

2 Table of Contents

Executive Summary

Kirstie Bellman, Andreas Herkersdorf, Mike Hinchey, Christian Müller-Schloer, Hartmut Schmeck, Rolf Würtz 1

Overview of Talks

Points of Entry between OC design and traditional and advanced design methodologies
Kirstie Bellman 5

Evolutionary Algorithms to support the design of self-organising manufacturing systems
Jürgen Branke 5

Researching the Artificial Hormone System – Lessons Learned
Uwe Brinkschulte 6

Designing Open Swarming Systems for Dynamic Runtime Adaptation
Sven A. Brueckner 6

Automating Decision Making
Yuriy Brun 7

Self-configuration from a Machine-Learning Perspective
Wolfgang Konen 7

Adding Mission Aware Resilience to the Enterprise
Robert Laddaga 8

Run-Time System Design
Chris Landauer 8

The Art of Organic Programming and Ercatons
Falk Langhammer 9

The Six-Legged Walking Robot OSCAR as a Demonstrator for the Organic Robot Control Architecture ORCA
Erik Maehle 9

Organic Computing: From Design-time to Run-time
Christian Mueller-Schloer 10

Self-Organized Self-Improvement: Using Self-Directed Experimentation to Improve Models and Methods
Phyllis R. Nelson 11

A proposal how to combine bottom-up emergence and top-down control during runtime
Gabriele Peters 11

Engineering Proprioception in Computing Systems
Marco Platzner 12

Cyber Physical Sytems (CPS) or better Cyber Biosphere(CBS)?
Franz J. Rammig 12

A Decade of experience building adaptive systems (Things that change in the night)
Paul Robertson 12

Overview of the DFG priority program Organic Computing
Hartmut Schmeck 13

Smart Grid, Renewables, Electric Mobility: Opportunities and Challenges for Organic ComputingC
Hartmut Schmeck 13

Physics-inspired Self-Organization and Adaptation in Large Dynamic Overlay Networks
Ingo Scholtes 14

Quantitative Emergence – An Overview of Recent Measurement Techniques
Bernhard Sick 14

Organic Self-organizing Bus-based Communication Systems (OrganicBus)
Jürgen Teich 15

Knowledge Representation for Autonomous Systems – The ASCENS Case Study
Emil Vassev 16

Slow Feature Analysis: Learning with the Slowness Principle
Laurenz Wiskott 17

Learning to see and understand
Rolf P. Wuertz 18

Self-adaptive workload management on MPSoC
Johannes Zeppenfeld 18

A glimpse of signaling pathways in the synapse
Junmei Zhu 19

Working Groups

Group I Report – Model-based Self-Adaptation 19

Group II Report – Learning and Context-dependent Dynamic Knowledge Representation 20

Group III Report – Dealing with Uncertainty in Organic Computing Systems . . . 24

Participants 28

3 Overview of Talks

3.1 Points of Entry between OC design and traditional and advanced design methodologies

Kirstie Bellman (Aerospace Corp. – Los Angeles, US)

OC has always been aware of the creative tension between the role and capabilities of the human developer and the role and capabilities of the OC processes. We have done an excellent job of building initial OC processes that allow us to research how to substitute for human design, especially, capabilities for a system to respond and self-adapt to an environment at run time. Now is a good time to reconsider broad new roles for OC within the development of complex systems from design through manufacturing.

This talk presents possible new "points of entry" between OC/self- organizational processes and advanced design methodologies. New model- based design-to-manufacturing processes include advancements in design generation, verification, complexity metrics, mathematically formal semantics and other good things. How could OC change/alter the design to manufacturing processes and yet coexist with or even leverage them? How could OC potentially improve the way a human being can interact with/manage a complex system (which includes the use of reflection, language, and collaboration?) The last part of the talk discusses the many challenges for both MBD and OC and suggests that they should be faced with complementary efforts.

3.2 Evolutionary Algorithms to support the design of self-organising manufacturing systems

Jürgen Branke (University of Warwick, GB)

Joint work of Branke, Jürgen; Pickardt Christoph
Main reference Branke, J. and Pickardt, C. Analysing job shop dispatching rules by automatically constructing difficult problem instances. European Journal Of Operational Research (2011)
URL http://www.informs-sim.org/wsc10papers/230.pdf

Abstract: Designing complex, self-organising systems is challenging. It requires to design local, decentralised rules for the agents which result in a good global performance of the overall system. In this talk, two approaches are presented at the example of a self-organising manufacturing system where local dispatching rules are used for decentralised scheduling.

The first approach supports a human designer by revealing the weaknesses of an examined manufacturing system. This is achieved by automatically searching for easy-to-analyse problem instances where the applied dispatching rule performs poorly.

The other approach is to generate the dispatching rules fully automatically by simulation-based Genetic Programming.

3.3 Researching the Artificial Hormone System – Lessons Learned

Uwe Brinkschulte (Universität Frankfurt am Main, DE)

The biological hormone system is a flexible and completely decentralized control mechanism. Therefore it is an excellent blueprint for technical self-organizing systems. We have implemented an artificial hormone system to control task assignment in distributed embedded systems. Hormones are emulated by short messages multicasted and broadcasted in the system. Three types of artificial hormones, eagervalues, suppressors and accelerators are used here. To prove the concept, a hormone simulator has been developed. Furthermore, a hormone based middleware has been implemented. Using self-synchronization mechanisms, the implementation and memory needs of hormone processing could be simplified significantly. Timing guarantees for task allocation could be improved from 2m to m cycles to allocate m tasks. It could be as well shown that the quality of task assignment by the artificial hormone system is better than pure load balancing. Furthermore, tight upper bounds for the communication load and stability conditions to avoid unbound task allocation could be derived. Open research questions are how to find good initial hormone levels in an automated way, how to protect the artificial hormone system against malicious attacks and how to apply the system to other fields of application.

3.4 Designing Open Swarming Systems for Dynamic Runtime Adaptation

Sven A. Brueckner (Jacobs Technology Inc. – Ann Arbor, US)

Traditional software methodologies place the burden of achieving an optimal system response onto the designer, hoping that any scenarios presented at runtime have been accounted for in the chosen optimization solution. In a world where systems are embedded in a dynamically changing environment and where system components have to act autonomously and without complete (or even correct) knowledge of the problem state, complete design-time optimization is no longer feasible. Instead, the role of the designer shifts from developing an optimal solution to developing a self-adaptive system that is capable of dynamically finding the appropriate solution at runtime.

At the seminar, we offer our experience in designing, implementing, and evaluating open self-organizing systems for real-world domains where the required capabilities (including self-adaptation for optimization) emerges from local interactions of many simple agents inspired by the architecture and processes of natural systems. In such open swarming systems, the agents are equipped to respond to changes in their environment to collectively reconfigure their activities for the emergence of optimal system-level patterns and functions. We present example designs from three application domains: dynamic prediction (the swarm induces optimal models of recently observed behavior to extrapolate them into the future), information management (the swarm rearranges models of document content and analyst interest to respond to new information or queries), and IED risk forecasting (the swarm dynamically accounts for new event patterns that may indicate shifts in insurgent operations). In each example, we discuss the critical design decisions that support dynamic self-organization for optimal operation at runtime.

3.5 Automating Decision Making

Yuriy Brun (University of Washington – Seattle, US)

Self-adaptive systems often evaluate potential adaptations via static model analysis or simulation. I propose an alternative evaluation scheme: trying the adaptation out either on the live, running system or in parallel to the un- adapted system, and observing the effects. This approach likely improves precision of adaptation evaluation, allows for quick adaptation implementation, and reduces costly adaptation undos, albeit, significant technical challenges remain.

3.6 Self-configuration from a Machine-Learning Perspective

Wolfgang Konen (FH Köln, DE)

Main reference W. Konen, "Self-configuration from a Machine-Learning Perspective," Dagstuhl Preprint Archive, arXiv:1105.1951v2
URL http://arxiv.org/abs/1105.1951v2

The goal of machine learning is to provide solutions which are trained by data or by experience coming from the environment. Many training algorithms exist and some brilliant successes were achieved. But even in structured environments for machine learning (e.g. data mining or board games), most applications beyond the level of toy problems need careful hand-tuning or human ingenuity (i.e. detection of interesting patterns) or both. We discuss several aspects how self-configuration can help to alleviate these problems.

One aspect is the self-configuration by tuning of algorithms, where recent advances have been made in the area of SPO (Sequential Parameter Optimization).

Another aspect is the self-configuration by pattern detection or feature construction. Forming multiple features (e.g. random boolean functions) and using algorithms (e.g. random forests) which easily digest many features can largely increase learning speed. However, a full-fledged theory of feature construction is not yet available and forms a current barrier in machine learning.

We discuss several ideas for systematic inclusion of feature construction. This may lead to partly self-configuring machine learning solutions which show robustness, flexibility and fast learning in potentially changing environments.

3.7 Adding Mission Aware Resilience to the Enterprise

Robert Laddaga (Doll Inc. – MA, US)

We envision technology to allow mission requirements to be expressed such that hosts, routers and media configuration can be tailored to those mission requirements in a semi-automated fashion. The mission requirements would accompany the configured system, and later be referred to as monitoring or human intervention indicated a change in requirements, or physical or cyber damage sustained by the system.

Regeneration or reconfiguration of components would then be enabled using the original or modified mission requirements, and the current monitored state of the system. We call the system "a mission-aware adaptive response system" (MARS). A MARS system would significantly improve mission effectiveness and cost, and potentially save lives.

Configuring to mission requirements plus whatever additional capabilities are mandated by policy will ensure that mission needs can be met, without expensive and unnecessary oversupply. More important is the effect of adaptation to mission changes or physical and cyber damage. Repair and reconfiguration could mean the difference between mission failure and success.

3.8 Run-Time System Design

Chris Landauer (Aerospace Corp. – Los Angeles, US)

In this paper, we describe an approach to building autonomous systems for hazardous environments that may be more flexible than the methods currently in use. The system gains this flexibility by doing some of its own system design in response to either environmental activity or internal failures or enhancements, at run-time. The basic idea is that it contains generic models of its own behavioral requirements, which are expected to interact with certain kinds of environmental behavior, and depend on certain capabilities of the hardware behavior. As the hardware degrades, or new software capabilities are provided, or the environmental behavior changes, the specializations used at the beginning of deployment are re- examined, and the decisions revisited. The approach depends on the Wrapping integration infrastructure for software-intensive systems, and on the expectation models for the evolution of the environment. All of the models can be changed remotely (as long as the necessary hardware is available), and the models define the behavior of the system.

3.9 The Art of Organic Programming and Ercatons

Falk Langhammer (Living Pages Research GmbH – München, DE)

Joint work of Langhammer, Falk; Imbusch, Oliver
Main reference Oliver Imbusch and Falk Langhammer and Guido von Walter, Ercatons and organic programming: say good-bye to planned economy, OOPSLA, p.41-52, 2005, Companion to the 20th Annual ACM SIGPLAN Conference on Object-Oriented Programming Systems Languages and Applications OOPSLA
URL http://portal.acm.org/citation.cfm?doid=1094855.1094868

Ercatons are the basic elements for a novel approach to (business) programming. They are simple, document-like building blocks with little a-priori constraints; but with all traditional object-oriented features except algorithmic code, added. Systems grow organically by adding and altering existing ercatons which may be done by other ercatons or humans and which happens at run-time. There is no formal design time. Complexity emerges from an increasing amount of interaction rather than code complication.

The above approach is in industrial production at a few corporations and was invented by us. It is a substitute for conventional and expensive enterprise programming.

The talk highlights the relationship with organic computing and lessons learned. A system grown by organic programming is an organic computing system if programmers and users are considered to be part of the system. Ercatons then enable the emergence of unforeseen features in such systems.

3.10 The Six-Legged Walking Robot OSCAR as a Demonstrator for the Organic Robot Control Architecture ORCA

Erik Maehle (Universität Lübeck, DE)

Joint work of Maehle, Erik; Brockmann, Werner; Grosspietsch, Karl-Erwin; Al-Homsy, Ahmad; El-Sayed-Auf, Adam; Jakimovski, Bojan; Krannich, Stephan; Litza, Marek; Maas, Raphael

Autonomous mobile robots are complex machines being challenging to engineer and to program. In order to master this complexity, the Organic Robot Control Architecture ORCA has been developed making use of organic principles like self-organization, self-reconfiguration and self-healing. The six-legged walking robot OSCAR is introduced as a demonstrator for ORCA. Organic principles are employed on all layers of its hierarchical control system starting at the reflexive layer with gait generation and reflexes over the reactive behavioural layer up to the deliberative planning layer. Experimental evaluations demonstrate that OSCAR thus attempts to continue its mission in the best still possible way by adapting to internal faults as well as to unforeseen environmental situations.

3.11 Organic Computing: From Design-time to Run-time

Christian Mueller-Schloer (Leibniz Universität Hannover, DE)

In 2003, when we discussed OC for the first time, we defined it as a technology, which brings life-like properties into complex technical systems to make them more robust and flexible. OC systems are designed to be self- organized, self-configuring, self-protecting, self-explaining etc.

Now, 8 years down the road, I want to suggest a new definition of OC: OC means to move design-time decisions to run-time. This definition has the advantage that it can be operationalized: We can derive the future OC research program if we follow the steps of classical design processes and ask for each of them what it would mean to move it into run-time.

This talk will sketch out this idea in more detail. First we will briefly review design processes and ex-tract a simplified version of a standard design process.

Next we will transfer this process to run-time arriving at a so-called "seamless redesign process". Now we can identify the main challenges that have to be met.

1. Shift from design-space to configuration space: It is the task of the designer to explore the possibilities of the design space. This is an activity, which requires experience as well as creativity. In an OC system the agent itself has to explore the configuration space. Hence it must be endowed with creativity and experience as well. The configuration space will have to be smaller than the design space in order to bound search-time. Also it should exclude illegal configurations in order to keep the creative agent from proposing configurations, which cannot be realized.
2. Run-time optimization: The OC system can be modeled as a cognitive agent moving through the configuration space in search of the highest fitness.
   This fitness landscape is not only unknown to the agent but also time- dependent and self-referential. i.e. it changes depending on the actions of the agents.
3. Run-time validation: An agent who searches the configuration space has to validate possible solutions in order to sort out detrimental ones and to find the best possible one. Validation could be done by trial-and-error but this means that bad (or illegal) solutions are tried out in reality. This is clearly not acceptable in technical systems. Therefore we have to introduce a sandbox approach where solutions are validated in a simulated environment. Simulation (and verification) is usually time consuming, and it requires an accurate description of the system. Moreover it re-quires stimuli reflecting the future real situation as closely as possible. Finally, we should consider the option of on-line testing.
4. Production vs. run-time reconfiguration: While in the classical design process, at a certain point in time the design is frozen and goes into production, the run-time reconfiguration process is seamless: This means in principle that all, even the higher-level, design decision can be revised. But it also means that only those changes can be actuated, which can be realized in terms of software or hardware reconfigurations.
5. Run-time modeling: All design steps, be it in the classical design process or in the run-time re-configuration process, are model-based. But now we have to consider two different flavors of the-se models: Prescriptive models reflect the classical top-down enforcement. Descriptive models reflect the actual system state. They are not necessarily consistent. We have to find mechanisms, which can minimize the possible contradictions.

This corresponds to a run-time version of what in the classical design process is called Yoyo design.

The talk closes with some remarks on more OC challenges, which have not been sufficiently ad-dressed so far: (1) The relationship between the user and the adaptive system, and (2) the broader view of regarding OC as a part of "Organizational Sciences".

3.12 Self-Organized Self-Improvement: Using Self-Directed Experimentation to Improve Models and Methods

Phyllis R. Nelson (Cal Poly – Pomona, US)

The models and methods that a system uses to choose its behaviors are an important link between the system's current situation and its top-level purposes and goals. If the system lacks models and methods that match its current condition and context, it may no longer be able to appropriately link its resources to its purposes and goals. Biological systems use some of their spare time and energy to explore both their own capabilities and their environment.

We discuss our version of this biological style for self-organizing behaviors that enable a system to improve its models of both its own current capabilities and its environment. We also consider how the system can determine if such self improvement is useful. Finally, we present a problem for this concept (the existence of a hidden variable), and examine the consequences for self organization.

3.13 A proposal how to combine bottom-up emergence and top-down control during runtime

Gabriele Peters (FernUniversität in Hagen, DE)

Joint work of Leopold, Thomas; Kern-Isberner, Gabriele; Peters, Gabriele
Main reference T. Leopold, G. Kern-Isberner, G. Peters, "Belief Revision with Reinforcement Learning for Interactive Object Recognition," Proc. 18th European Conf. on Artificial Intelligence (ECAI 2008), Vol. 178, pp. 65–69, Patras, Greece, July 21-25, 2008.
URL http://dx.doi.org/10.3233/978-1-58603-891-5-65

A system is proposed which combines two levels of learning. Both levels negotiate during runtime and establish a balance between bottom-up emergence and top-down control. They are realized by techniques of reinforcement learning (RL) and belief revision (BR). The RL component is able to react to runtime events and learns behavioral strategies in a flexible way. What is learned by RL is available in a numerical form only. Especially, it is not intuitively understandable by humans. In contrast, the BR component acquires knowledge in the form of rules. These rules control the bottom-up RL process from top-down. They are comprehensible by humans. Thus, the BR component can act as an interface for intervention from outside the system by a system designer or a user in case the system displays undesired behavior. Work in progress is presented where this system design is applied to a computer vision task.

References

1 Thomas Leopold, Gabriele Kern-Isberner, and Gabriele Peters. *Combining Reinforcement Learning and Belief Revision – A Learning System for Active Vision*. Proc. 19th British Machine Vision Conf. (BMVC 2008), Vol. 1, pp. 473–482, Leeds, UK, 2008

3.14 Engineering Proprioception in Computing Systems

Marco Platzner (Universität Paderborn, DE)

In this presentation I will give an introduction to the objectives and working areas of the recently started project "Engineering Proprioception in Computing Systems" (EPiCS). EPiCS is part of the EU FET objective Self-awareness in Autonomic Systems and relies on self-awareness and self-expression as key concepts for enabling complex future computing and communication systems.

Key words: Self-awareness, self-expression, autonomic systems

3.15 Cyber Physical Sytems (CPS) or better Cyber Biosphere(CBS)?

Franz J. Rammig (C-LAB – Paderborn, DE)

As most technical artifacts will be interconnected in some sense ("Internet of Things, Cyber Physical Systems"), IT systems of the future cannot be treated as isolated entities any longer. More or less every technical artifact will be linked to the Internet. Two major tendencies can be observed. The first one takes its inspiration from the technical roots of Embedded Systems. This approach became well known under the name "Cyber Physical Systems (CPS)". The main challenge of this approach is the necessity to bridge two incompatible worlds: this one of highly predictable embedded real-time systems and this one of the stochastically operated Internet. The second approach takes inspirations from the achievements of nature. This approach is being discussed using terms like "Biologically Inspired Systems" or "Organic Computing". We discuss these alternatives building the highly sophisticated Embedded Systems of the future.

The basic challenges to be solved when designing CPS as well as CBS are characterized. Some comparisons of CPS and CBS will be made as well.

3.16 A Decade of experience building adaptive systems (Things that change in the night)

Paul Robertson (Doll Inc. – MA, US)

In today's world computers are constantly connected and participating in the full breadth of human existence. Our world is changing at record breaking speed and yet our software is still largely designed and implemented as it was when programs were static, disconnected, and run on demand by people. My colleagues and I have now been building self-adaptive systems for over a decade and have applied them to some of the most challenging environments including real-time vision, robotics, and more recently to cyber security – where the world can change in real-time and the need to adapt is most crucial. A lot has been learned over that period about successful approaches, technologies, and research challenges.

3.17 Overview of the DFG priority program Organic Computing

Hartmut Schmeck (KIT – Karlsruhe Institute of Technology, DE)

Joint work of Christian Müller-Schloer; Schmeck, Hartmut; Ungerer, Theo

Originating from a series of workshops on future research topics for Computer Engineering back in 2002, a joint position paper of the Gesellschaft für Informatik and the Informationstechnische Gesellschaft outlined the vision of Organic Computing Systems to cope with the increasing presence of intelligent, interacting devices in various application scenarios by controlled self-organization and an emphasis on realizing robust, adaptive and trustworthy systems showing an "organic" behavior even in unanticipated situations. The talk provides a brief survey of the priority program on Organic Computing which has been funded by the German Research Foundation (DFG) from 2005 to 2011. The results of the program have been compiled into a comprehensive compendium on Organic Computing (see [MSU11]) with chapters on Theoretical Foundations, Methods and Tools, Learning, Architectures, Applications, and an Outlook on recently added projects and potential future directions of research.

References

1 Christian Müller-Schloer, Hartmut Schmeck, Theo Ungerer (eds.) *Organic Computing – A Paradigm Shift for Complex Systems*. Birkhäuser Verlag, Basel, Schweiz, 2011

3.18 Smart Grid, Renewables, Electric Mobility: Opportunities and Challenges for Organic ComputingC

Hartmut Schmeck (KIT – Karlsruhe Institute of Technology, DE)

The strong shift in energy policy towards power generation from renewable sources leads to a number of challenges for an adequate management of the future power grid. In particular, the uncontrollable and only partially predictable fluctuations in energy supply ask for a replacement of the traditional paradigm "supply follows demand" with the new principle "demand follows supply" in addition to a shift from centralized power generation to large scale decentralized supply of power at the low voltage segments of the distribution grid. This relies on detailed information on the current status of the relevant components of the distribution grid and on sufficient knowledge about the available degrees of freedom for demand shifting. Organic Computing has the potential to provide adequate concepts for shaping these new approaches to energy management which should combine electrical and thermal energy. The talk outlines the various individual challenges and describes some experiences in the projects MeRegio and MeRegioMobile, where a smart home has been designed for investigating the intelligent integration of the charging needs of electric vehicles into an environment containing a range of different consumers and suppliers of energy.

3.19 Physics-inspired Self-Organization and Adaptation in Large Dynamic Overlay Networks

Ingo Scholtes (Universität Trier, DE)

Overlay networks are becoming an increasingly important abstraction that facilitates the cost-efficient and scalable provision of novel services in the Internet. However, efficiently constructing, maintaining and managing robust and adaptive overlay topologies in the face of highly dynamic participants is a challenging task.

In this talk, a physics-inspired approach towards the management of self- organizing and self-adaptive overlays will be discussed. It is based on the idea that global-scale network infrastructures like the Internet or our biggest Peer-to-Peer systems are becoming so large that it appears justified to design them along models and abstractions originally developed for the study of many-particle systems in statistical physics. The management schemes that will be presented take advantage of recently uncovered analogies between random graph theory and statistical mechanics. They constitute the basis for what may be called a thermodynamic management of large dynamic overlay networks.

3.20 Quantitative Emergence – An Overview of Recent Measurement Techniques

Bernhard Sick (Universität Kassel, DE)

Joint work of Fisch, Dominik; Jänicke, Martin; Kalkowski, Edgar; Sick, Bernhard

A technical system exhibits emergence when it has certain properties or qualities that can be termed to be irreducible in the sense that they are not traceable down to the constituent parts of the system. The presentation summarizes three techniques for emergence detection and emergence measurement that were proposed by members of the organic computing community. These techniques are based on information- theoretic and probabilistic viewpoints: the discrete entropy difference, the Hellinger distance which is a divergence measure for probability densities, and an iterative approach motivated by divergence measures. Advantages and drawbacks of these measures are demonstrated by means of some simulation experiments using artificial data sets. It is shown that these techniques are able to deal with different kinds of emergent phenomena such as transitions from chaos to order, concept drift, or novelty. That is, with these techniques it is possible to cover a wide range of possible applications. Moreover, it will be possible to build systems that are self- aware, environment-aware, self-reflecting, ...

3.21 Organic Self-organizing Bus-based Communication Systems (OrganicBus)

Jürgen Teich (Universität Erlangen-Nürnberg, DE)

Joint work of Teich, Jürgen; Ziermann, Tobias; Wildermann, Stefan
URL http://www12.informatik.uni-erlangen.de/research/organicbus/

We present an organic computing approach for the analysis, design and optimization of run-time message scheduling for priority-based bus systems such as the industrial CAN (Controller Area Network) standard.

The goal of this new approach is to overcome the major drawbacks of today's pure offline scheduling decisions that are based on worst-case assumptions, are not flexible or overly pessimistic with respect to unknown, uncertain or dynamically changing message request scenarios.

In contrast, our decentralized approach using online self-organization is able to monitor the actual traffic of the shared bus medium and adapt either sending rates, probabilities or offsets to establish fair bandwidth sharing and/or reduced response times.

For messages with high bandwidth demands such as streams, we present a decentral adaptation algorithm called *penalty learning algorithm* (PLA) [1] that is able to achieve a fair bandwidth assignment of a set of sending nodes and reaching this equal utilization provably no matter of the initial priority assignment by applying a game-theoretic analysis.

For periodic messages with (soft) real-time constraints as given by deadlines, we introduce a simple decentralized algorithm for adapting the offsets of message in a bursty time interval such that the average observervable worst case response time is reduced.

As a side effect of observing and manipulating this traffic for a monitor interval in the size of the hyperperiod for soft-real time periodic messages, a bus might be driven with higher utilization rates while still satisfying acceptable response times.

This algorithm called *Dynamic Offset Adaption Algorithm* (DynOAA) [2] for soft real-time tasks may be jointly run with the algorithm PLA for bandwidth type of messages.

In the near future, an implementation an FPGA-based platform coupled to a real CAN-bus shall give experimental evidence of the superiority of these techniques with respect to static scheduling.

References

1 Wildermann, Stefan and Ziermann, Tobias and Teich, Jürgen. *Self-organizing Bandwidth Sharing in Priority-based Medium Access.*
Proceedings of the Third IEEE International Conference on Self-Adaptive and Self-Organizing Systems (SASO 09), San Franzisco, USA, 2009
2 Ziermann, Tobias and Salcic, Zoran and Teich, Jürgen. *DynOAA – Dynamic Offset Adaptation Algorithm for Improving Response Times of CAN Systems.*
Proceedings of Design, Automation and Test in Europe (DATE), IEEE Computer Society, Grenoble, France, 2011

3.22 Knowledge Representation for Autonomous Systems – The ASCENS Case Study

Emil Vassev (University of Limerick, IE)

Joint work of Vassev, Emil; Mike Hinchey
Main reference E. Vassev, M. Hinchey, "Knowledge Representation and Awareness in Autonomic Service-Component Ensembles – State of the Art," Proc. 14th IEEE Int'l Symp. on Object/Component/Service-oriented Real-time Distributed Computing Workshops (ISCORCW'11), IEEE, 2011, pp.110–119.

Introduction

Ideally, autonomous systems are intelligent systems employing knowledge to become aware of situations, recognize changes and eventually respond to changing conditions. Knowledge is the key to such autonomous behavior. The fundamental questions are how to represent knowledge in such systems and how to make them use and manage that knowledge.

Current and ongoing research at Lero – the Irish Software Engineering Research Centre, is focused on the problem of knowledge representation for autonomous systems formed as ensembles of special autonomous service components. Such components encapsulate rules, constraints and mechanisms for self-adaptation and acquire and process knowledge about themselves, other service components and their environment. One of the expected major scientific contributions of this research is a formal approach to knowledge representation and reasoning mechanisms that help autonomous components acquire and structure comprehensive knowledge in such a way that it can be effectively and efficiently processed, so the system becomes aware of itself and its environment.

Autonomic Service-Component Ensembles (ASCENS) (see http://www.ascens-ist.eu/) is a class of multi-agent systems formed as mobile, intelligent and open-ended swarms of special autonomic service components capable of local and distributed reasoning. Such service components encapsulate rules, constraints and mechanisms for self- adaptation and acquire and process knowledge about themselves, other service components and their environment. ASCENS systems pose distinct challenges for knowledge representation languages.

Kinds of Knowledge for ASCENS

There have been determined four basic knowledge domains (kinds of knowledge) for ASCENS systems:
- the individual component structure and behavior;
- the system structure and behavior;
- the environment structure and behavior;
- situations where the system might end up in.

These knowledge domains have been used to derive distinct knowledge models (each representing a distinct knowledge domain) for ASCENS forming a high-level knowledge structure that is to be maintained by any service component (SC) member of a service-component ensemble (SCE):
- SC knowledge model – knowledge about internal configuration, resource usage, content, behavior, services, goals, communication ports, actions, events, metrics, etc.;
- SCE knowledge model – knowledge about the whole system, e.g., architecture topology, structure, system-level goals and services, behavior, communication links, public interfaces, etc.;

- environment knowledge model – parameters and properties of the operational environment, e.g., external systems, external communication interfaces, integration with other systems, etc.;
- situational knowledge patterns – specific situations, involving one or more SCs and eventually the environment.

By representing the knowledge in such models, an individual SC shall be able to query information about both the SC itself and the SCE, by considering the environment's parameters and properties. Moreover, this helps SCs understand and reason about themselves and discover situations through the use of probabilistic methods working over the knowledge modeled as situational knowledge patterns.

3. Structure of the Knowledge Representation The four knowledge domains for ASCENS are represented by four distinct knowledge corpuses – SC Knowledge Corpus, SCE Knowledge Corpus, Environment Knowledge Corpus and Situational Knowledge Corpus. Each knowledge corpus is structured into a special domain-specific ontology and a logical framework. The domain-specific ontology gives a formal and declarative representation of the knowledge domain in terms of explicitly described domain concepts, individuals (or objects) and the relationships between those concepts/individuals. The logical framework helps to realize the explicit representation of particular and general factual knowledge, in terms of predicates, names, connectives, quantifiers, and identity. Thus, the logical framework provides additional to the domain ontology computational structures that basically determine the logical foundations helping a SC reason and infer knowledge.

All the four ASCENS knowledge corpuses form together the ASCENS Knowledge Base (AKB). The AKB is a sort of knowledge database where knowledge is stored, retrieved and updated. Therefore, in addition to the knowledge corpuses, the AKB implies a knowledge-operating mechanism providing for knowledge storing, updating and retrieval/querying. Ideally, we can think of an AKB as a black box whose interface consists of two methods called TELL and ASK. TELL is used to add new sentences to the knowledge base and ASK can be used to query information. Both methods may involve knowledge inference and therefore, an AKB should be equipped with a special Inference Engine that reasons about the information in the knowledge base for the ultimate purpose of formulating new conclusions, i.e., inferring new knowledge.

3.23 Slow Feature Analysis: Learning with the Slowness Principle

Laurenz Wiskott (Ruhr-Universität Bochum, DE)

Joint work of Wiskott, Laurenz; Berkes, Pietro; Franzius, Mathias; Sprekeler, Henning; Wilbert, Niko; Zito, Tiziano
Main reference Wiskott, L.; Berkes, P.; Franzius, M.; Sprekeler, H.; Wilbert, N. (2011) Slow feature analysis. Scholarpedia, 6, 5282
URL http://www.scholarpedia.org/article/Slow_feature_analysis/

Slow feature analysis (SFA) is a biologically motivated algorithm for extracting slowly varying features from a quickly varying signal and has proven to be a powerfull general-purpose prepocessing method for spatio-temporal data. We have applied SFA to the learning of complex cell receptive fields, visual invariances for whole objects, and place cells in the hippocampus. On the technical side SFA can be used to extract slowly varying driving forces of dynamical systems and to perform nonlinear blind source separation. Here I will introduce the SFA algorithm and give an overview over these different applications.

3.24 Learning to see and understand

Rolf P. Wuertz (Ruhr-Universität Bochum, DE)

Joint work of Wuertz, Rolf P.; Müller, Marco K.; Walther, T.

Sensor interpretation is one of the central requirements for computing systems to interact sensibly with their environment. Moreover, the transformation of sensor data into semantically meaningful representations is necessary to exploit the types of computation in which machines are much better than humans.

The most important among the sensory modalities is visual processing. In this talk I reviewed self-organizing methods we have developed for the recognition of human faces and human bodies.

The first system [1] is based on similarity rank lists and can learn the transformation from a frontal face to a different pose strictly on the basis of examples. This can be carried out by a neural network in a natural way. It also presents a way of controlling the generalization of the recognitition system.

The second system [2] uses several state-of-the-art methods from computer vision to learn a model of the appearance and possible kinematics of the upper body of humans from videos in an unsuupervised way.

The resulting model is able to generalize over individuals, clothing, and a wide range of backgrounds, and is robust enough for still image interpretation.

References

1 Marco K. Müller and Rolf P. Würtz. Learning from examples to recognize faces in different situations. *Neural Networks*. Submitted.
2 Thomas Walther and Rolf P. Würtz. Autonomous learning of a human body model. In *Proc. IJCNN*. IEEE, 2011. In press.

3.25 Self-adaptive workload management on MPSoC

Johannes Zeppenfeld (TU München, DE)

Joint work of Zeppenfeld, Johannes; Bouajila, Abdelmajid; Stechele, Walter; Herkersdorf, Andreas
Main reference J. Zeppenfeld, A. Herkersdorf, "Applying Autonomic Principles for Workload Management in Multi-Core Systems on Chip," Int'l Conf. Autonomic Computing (ICAC'11), Karlsruhe, Germany, June 14–18, 2011

Due to the difficulty involved in designing ever more complex systems, a trend is developing to simply replicate existing hardware components, rather than extending the functional capabilities of existing components. While this simplifies the work of the hardware designers, it simply offloads the difficulty of designing complex systems to the software developers, who must somehow make use of all the resulting parallel processing units.

The Autonomic System on Chip paradigm suggests an alternative approach, namely by utilizing a portion of the gained resources for the addition of bio- inspired enhancements, which can autonomously accomplish at run time many of the tasks previously performed by the designer at design time. Not only does this reduce the burden on the designer, it also allows the system to adapt to unforseeable environment- or system-states. The presented work shows how such autonomic principles, applied to a general purpose multi-core system on chip, can autonomically adjust the frequencies and distribute tasks across the available processing cores.

3.26 A glimpse of signaling pathways in the synapse

Junmei Zhu (EBI – Cambridge, GB)

Learning is a key to run-time design. One of the most studied learning elements is the synapse. Computationally, Hebbian plasticity can be conveniently described by one equation. In biology, however, thousands of genes have been identified in a typical synapse. The underlying molecular mechanism for plasticity seems to be escaping our grasp, with an ever-expanding list of involved molecules. I will introduce this picture and our effort to construct the complete signalling pathways. The complexity could be what is needed for the robustness and flexibility of organic systems, and thus OC design is not to be intimidated by large systems.

4 Working Groups

4.1 Group I Report – Model-based Self-Adaptation

4.1.1 Challenges

We identified five challenges related to model-based self-adaptation:
1. Defining what a model is and establishing a consistent, clear terminology. Models can be of the system itself and of the environment.
2. Selecting the right model: abstraction level, objective function, time scales.
3. Understanding control-loop design and making control-loop development explicit during the design process.
4. Identifying those environments that can benefit most from self-adaptation (as well as those that cannot benefit or will present significant challenges). One aspect of these environments is exploration vs. exploitation and competition for resources.
5. Reducing the need for the design-time. This challenge aims to develop some framework that will let the developer specify a high-level system goal and then have the system design itself (perhaps evolve) into one that satisfies that goal. We believe that breaking up the target system into a set of tiers may be a fruitful approach. Tiers are made up of building blocks. Each tier can "rearrange" its building blocks in order to achieve it's goal. Each tier also sets the goals for its building blocks, which are themselves the next tier. For example, a 3-tier traffic light system may consist of:
   - Neighborhood (top tier): many adjacent intersections
   - Single intersection (middle tier): four traffic lights at one intersection
   - Single traffic light (bottom tier).

The developer would specify the high-level goal to the top tier. For example, "minimize the average travel time through the neighborhood while avoiding accidents." That tier would then attempt to set a goal for the middle tier. e.g., "minimize the maximum waiting time for a car at the intersection while ensuring no intersecting car paths have the green light at the same time.” The top tier can (1) manipulate the intersections and (2) set the goals for the intersections. The intersections then do the same for the bottom tier: the traffic lights. The goals for the bottom tier may be “only one light on at a time, green $\rightarrow$ yellow $\rightarrow$ red $\rightarrow$ yellow $\rightarrow$ green . . . ”.

We believe that each tier must operate at a slower time scale than the tier below it. Possible techniques for manipulating the lower tier to accomplish a goal include: simulated annealing, artificial evolution, genetic algorithms. One concern is whether such exploration is likely to get stuck in local minima. Background exploration with one of the above techniques can help get out of such situations, but it is unlikely to consistently find the optimum solution. The design of the tiers constrains the search space. This is one way to constrain that search space, and we are not sure that better ways do not exist. We hope there can be tools made to automate, semi-automate, or otherwise help developers design these tiers.

4.2 Group II Report – Learning and Context-dependent Dynamic Knowledge Representation

4.2.1 Challenges

4.2.1.1 How can we LEARN emergent features on several layers / hierarchies?

(Gabriele Peters, Laurenz Wiskott) As one open question we identified the problem of getting knowledge into a system. If learning takes place in a hierarchical manner, then probably different methods for knowledge acquisition are needed for different levels of learning. Whereas on a lower level a statistical approach for learning features may be sufficient, this probably does not hold true for higher levels. On a low level features are relatively simple and low-dimensional and plenty of samples are available for training. Examples are small image patches or sound snippets. This makes purely statistical methods feasible. On a high level, features become more complex and high dimensional. Examples are whole visual objects or scenes. It is therefore not possible anymore to learn them by brute force statistical methods. Another reason why it might be good to have different mechanisms on a high level is that the system needs to do reasoning about things, which requires a more symbolic representation, at least in addition to a feature based representation (where a feature is understood here as something that has a graded value, while a symbol is binary in nature).

From, e.g., feature learning on the one end until scene understanding / interpretation on the other end there has to be one point in the hierarchy where the acquired knowledge has to be transferred from an implicit / numerical form to a more explicit / symbolic representation that allows for interpretation and reasoning. One possibility would be a symbolic representation in the form of rules. Gabriele Peters has shown in her talk that it is possible that such rules can be learned “organically” during runtime [1]. On a lower level an implicit representation of appropriate behavior is learned by reinforcement learning; on a higher level rules are generated and checked against the reinforcement model to formalize and make explicit what has been learned implicitly on the lower level. The symbolic and rule based representation on the higher levels permits reasoning, planing, and anticipation as needed for the WHAT IF strategy, see below.

We also discussed the question whether one could replace the structured learning on the higher levels by pure memory. Besides the fact that any system has a limited memory capacity that would quickly be filled with sensory data, are there any other advantages for not storing everything and then working on the memories. It seems clear that working on the collection of all memories not only raises the problem of limited memory capacity but also poses a problem for other resources, e.g. computational resources. For instance, a nearest neighbor classifier becomes quite inefficient, if too many samples are stored. Thus, it is also computational limitations that prevent a system from working on complete memory collections. Another problem is the lack of interpolation and regularization. If data is noisy and/or sparse the system needs to perform some regularization in order to learn appropriate functionality that generalizes well. This could, of course be done on the fly each time the functionality is needed, but this would be computationally even more expensive.

Thus, the need for a condensed and explicit/symbolic representation of the world on higher layers, i.e. the need for an efficient model of the world, results from memory limitation, computational limitations and the need for planing and reasoning. Quasi symbolic representations can also be derived from single exemplars given the right representation, see Section "How can we learn from few examples".

(Phyllis R. Nelson) There is also the question of how to start the system. What knowledge, models, methods and procedures should the designers put in? Should this set of initial resources be privileged in some way to ensure that the system can always be restarted? For any complex system, the designers don't know enough about some aspects of the system (otherwise we wouldn't need OC), and may also know so much from their own experience that the initial knowledge must be restructured by the system.

References

1 Thomas Leopold, Gabriele Kern-Isberner, and Gabriele Peters, Combining Reinforcement Learning and Belief Revision – A Learning System for Active Vision. 19th British Machine Vision Conference (BMVC 2008), Vol. 1, pp. 473–482, 2008

4.2.1.2 Knowlege organization: How to not get stuck in too many interconnections between elements?

(Christoph Landauer) We have proved [LB98] that any system of knowledge with any construction mechanism will "get stuck"; eventually, each new item needs to be connected with so many other items that it cannot be effectively entered, even by humans. Circumventing this problem is an important issue in knowledge representation, and we think that it will require a very different approach to representation of knowledge.

[LB98] Christopher Landauer, Kirstie L. Bellman, "Situation Assessment via Computational Semiotics," pp. 712-717 in Proceedings of ISAS'98: The 1998 International MultiDisciplinary Conference on Intelligent Systems and Semiotics, 14–17 September 1998, NIST, Gaithersburg, Maryland (1998)

(Emil Vassev) Knowledge should be presented at different levels of abstraction and conceptually organized. Different abstract levels shall form levels of generality (introducing certain degree of uncertainty though) that can be applied to reasoning algorithms to release heavy computations by moving from a low-level knowledge to a more generic one.

The concept-based representation is necessary, because this is probably one of the most efficient ways to give both meaning and semantics to the raw data. Ontologies and description logics (as one of the most prominent ways to write ontologies) may help not to have unnecessary interconnections between the knowledge elements. An ontology is open-ended and should not include all the possible conceptual descriptions of the context, but only the relevant ones, i.e., giving sufficient level of description taking into consideration the operational domain of the system in question. Moreover, techniques like subsumtion shall help to make new generalizations over the concepts, thus reducing the amount of concepts that must be processes when the system reasons. For example, subsumtion may help to exploit the taxonomy structure of concepts that are defined in the ontology and compute a new taxonomy for a set of concepts.

4.2.1.3 Finding the relevant knowledge in a given context (contect might change on-line)

(Christoph Landauer) This issue is about how a system uses knowledge that it has. Whatever knowledge scheme is used, only a tiny fraction of that knowledge is appropriate for any given situation, and we need to have systems that can separate and isolate that relevant knowledge in real time.

(Wolfgang Konen) Another way to phrase it is that in a future OC system there might be a large feature base where certain sets of features are activated / deactivated with a mechanism yet-to-be-found depending on the context / the environment situation. The challenge is here to find mechanisms which are able to work in many relevant situations.

(Phyllis R. Nelson) The WIM and play (exploration) are a tool for finding out about those special contexts that are at the boundaries of the capabilities of the system. For a system with physical parts, playing games provides a structure for examining potentially "dangerous" behaviors and learning better models of those actions before they are invoked in "real" responses. In a game, the system can start and and then back away from these regions of operation since they are not directly part of accomplishing its main goals and purposes. This type of exploration is particularly important because the limits of physical subsystems are often nonlinear and depend on the history of the components (wear, old batteries, etc.).

4.2.1.4 Parallel knowledge representations (playing piano): how to merge / how to make that they support each other

(Christoph Landauer) The question here is how to merge suggestions from different kinds of knowledge in a seamless way. For the piano example, it is a combination of knowledge of the way the music looks on the score, knowledge of the way the music should sound, muscle knowledge of the way to move to make the sounds (this is why you practice), and knowledge of the way the music sounds just now. These different sources of very different kinds of knowledge are combined to make a sequence of actions that play the piece. In the piano example the different representations may support each other to perform the complex motor actions and memory retrievals. The hard part in general is blending the different knowledge sources, deciding if there is a conflict among the sources, resolving that conflict in real time, using whatever source is available to determine the next actions, and making all the relevant decisions very quickly.

(Phyllis R. Nelson) There is also the question of selecting the variables or features that will be measured in order to "close the loop" when the system acts. Producing the desired behaviors requires knowledge and models of the available sensing and actuation and how those variables that can be measured relate to the desired behavior. Play (exploration) is a way to build these connections.

4.2.2 Other points

4.2.2.1 How to learn from few examples?

(Rolf P. Würtz, Gabriele Peters, Laurenz Wiskott) Gradient-based learning methods usually require the iterated presentation of a large number of training examples, which makes it

comparably slow. Given an appropriate format, however, a single example may be sufficient to "know" the visual example. This works perfectly in human vision and nicely in face recognition systems. On a symbolic level, single examples can easily prove (exist) or disprove (forall) rules. If rules have a measure of evidence, this can be updated by every example. In G. Peter's system, each new example is able to qualitatively modify the behavior. The hierarchical SFA system by Laurenz Wiskott learns invariant representations from the presentation of few examples. Thus, learning from few examples is both feasible and desirable, especially for online real-time learning. *Christopher Landauer)* agree, when we decide that parallel performance improvement methods must be used, we can take into account the local structure of problems. An early example of learning symmetries from only one example was given by Konen and von der Malsburg in *Konen, W., von der Malsburg, C., "Learning to generalize from single examples in the dynamic link architecture," Neural Computation, 5, 1993, p. 719-735.*

4.2.2.2 Humans have a very elaborate What-If-mechanism (WIM)

(Wolfgang Konen, Laurenz Wiskott) Humans are good at combining prior experience and facts from new a environment in "What-If-simulations". The mechanism on how we can set up and utilize this very flexible mechanism is not yet fully understood and poses a challenge to be captured by future OC systems. But it is clear that "What-If-simulations" are a powerful tool to extend and recombine existing knowledge on the one hand and to save resources and avoid risks on the other hand.

(Laurenz Wiskott) However, What-If simulations have at least two types of limitations. If the world model is inaccurate, What-If simulations are bound to become unrealistic and therefore useless with increasing depth. This makes What-If simulations useful only up to a certain point in the future and then one has to take the real action in order to verify the predictions of the model. If the world model is perfect, What-If simulations can in principle substitute completely for the real exploration. Learning how to play a game by simulated self-play is a good example of WIM, but also of exploration. Perhaps this approach is useful in order to help limit the choices enough so that useful and relevant hypotheses can be inferred. However, in many cases this might be limited by memory capacity. For a human, for instance, it would be impossible (with a few exceptions, maybe) to learn to play chess by just imagining games against oneself, simply because it is so hard to keep the positions in mind accurately during an imagined game.

(Laurenz Wiskott) Thus, real exploration is needed if the model is inaccurate or if memory limitations prevent the system from What-If simulations of sufficient depth. A third reason for real exploration is, of course, if there is no adequate model for that particular domain, which however could be considered an extreme case of model inaccuracy. In [1] for instance a system is proposed where a model is learned from scratch by exploration. It is clear that the What-If simulations give strong hints as to where real exploration might pay off most. For instance, a good model should also maintain an estimate for the accuracy of its predictions. If predictions become uncertain, then exploring this part to improve the model is probably particularly advantageous.

References

1 Thomas Leopold, Gabriele Kern-Isberner, and Gabriele Peters. Belief Revision with Reinforcement Learning for Interactive Object Recognition. 18th European Conference on Artificial Intelligence (ECAI 2008), Vol. 178, pp. 65-69, 2008

4.2.2.3 How to cope with the combinatorial complexity of world in our brain model

(Laurenz Wiskott) The world is tremendously complex overall, but it is not possible to integrate all that in a model to begin with. Thus, one has to employ different mechanisms to start simple and grow complex gradually. During this process the degrees of freedom of a model should always be smaller than the available data that constrain it, in order to avoid overfitting and guarantee good generalization. To do so, it is good the start with a simple model, to which more complexity can be added as more data becomes available and more complexity is needed to explain it. When starting with such a simple model, it might be advantageous to first confront it only with part of the data, which allows for simplified modeling. When learning a language, for instance, start first with short sentences. Once theses are mastered by the model proceed to longer and more complex sentences, see [1]. This can be done because the world is highly structured and not just complex. There are simple parts that can be modeled largely independently of the rest and still one can use it to generalize to more complex domains. Or one can approximate complex domains by simple models and still get useful predictions out of it.

(Wolfgang Konen) An open issue is how to cope with the combinatorial complexity of the environment and it might call for knowledge representation models which incorporate also a similar way of combinatorial complexity.

References

1 Jeffery L. Elman. Learning and development in neural networks: the importance of starting small. *Cognition*, Vol. 48, pp. 71–99, 1993.

4.3 Group III Report – Dealing with Uncertainty in Organic Computing Systems

Being able to deal with uncertainty is a central concept that permeates many – if not most – aspects of Organic Computing systems, including run-time adaptation, knowledge representation, modeling, learning or feature extraction. In fact, one of the major promises of Organic Computing Systems – and one that potentially distinguishes them from traditionally engineered systems – is that they shall be able to deal with unanticipated situations, surprises and rare events. Furthermore, Organic Computing systems need to adapt and evolve at run-time, thus potentially introducing additional uncertainty with respect to their run-time behavior. In our group discussion we first tried to characterize different sources of uncertainty in Organic Computing Systems.

4.3.1 Environmental Uncertainty

One of the primary sources of uncertainty is the environment into which our computing systems are embedded. Here, uncertainty may result for example from technical failures, attacks, changing environmental conditions or human behavior. In most cases, one cannot actively control these environmental conditions, one can merely try to characterize the resulting uncertainties and reason about them either in a qualitative or in a quantitative fashion. And even this constitutes a considerable challenge. When trying to reason for example about the probabilities of certain events, one must deal with the fact that we typically don't even know the probability space to begin with, that is we don't know in advance which events may possibly occur, let alone their probabilities. In traditionally engineered systems, we are frequently reminded of this fact by the occurrence of unanticipated events. When dealing with rare events – like for instance catastrophic earthquakes – we also need to face the challenge that there are – if any – only few prior instances from which we could learn about the probability of these events. Here, a modeling in terms of extreme value theory seems to be crucial. Also, second order probability techniques can often be reasonably applied to characterize and measure a system's uncertainty about its environment.

In the discussion above, we inherently assumed that environmental uncertainty results from a lack of knowledge which can be reduced by observation, experimentation and probabilistic reasoning. Whether the fundamental reason for uncertainty is a mere lack of knowledge about the system's details or whether there are processes that are inherently random is a question that is still being discussed in fundamental physics and philosophy. In fact, we can even find different interpretations of uncertainty when considering frequentistic or Bayesian notions of probability in mathematics. Independent of this philosophical question, in practice there clearly is a limit to how much we can reduce environmental uncertainty by means of experimentation and probabilistic reasoning. An important limiting factor is that in distributed systems our knowledge about the current state of a system is necessarily imperfect and incomplete and that many systems exhibit a sensitivity to initial conditions that tightly limits their predictability.

While we typically embrace self-organization as a crucial concept in the design of Organic Computing systems, self-organization processes taking place in the environment are a frequent source of uncertainty and surprising effects. In fact, simple probabilistic models about the behavior of human, social and technical systems often do not account for complex collective behavior and correlations that result from the complex and often very subtle interactions between individual elements. This often spontaneously occurs at certain critical points in a system's parameter space, thus considerably hindering a sound reasoning about a system's behavior.

4.3.2 Operational Uncertainty

A different kind of uncertainty in Organic Computing systems may be called operational uncertainty. In contrast to environmental uncertainty, here we refer to the use of probabilistic approaches that deliberately introduce an uncertainty about the system's exact behavior that can nevertheless be controlled, adapted and reasoned about. In particular, the active and controlled use of probabilistic schemes is an important approach in the design of systems that incorporate a degree of variation and permissiveness, thus introducing a range of variability that is the source of self-adaptation and self-optimization in Organic Computing Systems. The importance of randomness and variability for is particularly visible in evolutionary algorithms, particle swarm optimization and simulated annealing techniques that are being regularly applied in the design of self-adaptive systems.

Two further aspects of a deliberate and meaningful introduction of randomness into a system can be related to the order-from-noise principle that has been proposed by Heinz von Foerster. First of all, the sensible introduction of randomness or noise into a system can paradoxically result in self-organization processes producing patterns and structures that are more stable and predictable. However, for the degree of noise there typically is a critical point above which noise hinders the self-organization of structures and patterns. As such, the deliberate and meaningful introduction of randomness into a system can be a powerful tool both to foster beneficial self-organization as well as fight the self-organized formation of unwanted patterns and correlations. So one might actually be tempted to say that it is often reasonable to fight uncontrolled uncertainty by deliberately introducing uncertainty in a controlled fashion, thus facilitating a sound stochastic reasoning about systems. A particularly simple illustrating example for this kind of approach in traditional algorithm engineering is the randomization of input sequences in order to facilitate a sound reasoning about the performance of sort algorithms like QuickSort (for instance in order to get rid of correlations in the input that might result in worst-case performance). Similarly, in Organic Computing systems randomization can be a very useful technique to get rid of unwanted correlations that potentially threaten the system's functioning and performance.

As suggested by the idea of fighting uncontrolled uncertainty by introducing controlled uncertainty, there is an intimate relationship between environmental and operational uncertainty. In fact, we may view the run-time adaptation of systems as as process that matches these two.

4.3.3 Designing Systems for Uncertainty

Having characterized different kinds of uncertainty as well as their importance for Organic Computing systems, we discussed different approaches that promise to improve the handling of unforeseen and rare events in practical systems.

4.3.3.1 Coping with everyday operations

First of all, simply due to the rarity and magnitude of extremal events, it often seems to be a best-practice approach to explicitly distinguish between everyday operations and rare, critical situations. One important reason for this distinction are for instance the often different requirements we may have. In everyday operations we are interested in the system's performance, cost-effectiveness and – at least to a certain degree – its optimality. In critical situations however, we are rather interested in the fact that the system maintains certain basic characteristics that are usually much more modest and relaxed. Under many circumstances it seems reasonable for systems to actively switch between two modes of operation.

4.3.3.2 Building fail-safe systems

In order to build fail-safe systems that are able to survive critical situations, we discussed the approach of goal-shedding. This is intimately related to the switch in the operational mode discussed above. The idea is to formulate a set of prioritized goals which the systems tried to achieve depending on the situation. In critical situations the system should be allowed to throw certain of these goals overboard in order to be enabled to meet at least the more important ones. Here, we discussed a practical example of a robotic submarine which is allowed to sacrifice crucial mission goals in order to at least safely return home based on the remaining power supply. Sometimes pursuing such goal-shedding strategies can require seemingly unintuitive behavior. Here we discussed the example of controlling cascading failures in complex networked systems. In certain classes of network topologies, it can actually be shown that intentionally killing almost all nodes in the system at the first indication of cascading faults is a good strategy to fight critical systemic faults. This admittedly drastic approach sacrifices most of the system's functionality but at the same time guarantees that the system can at least maintain a very basic service, while it would fail completely if no measures were taken. Preventing such errors can also facilitate the system to restart gracefully from a controlled state rather than being brought to a situation where nothing can be done anymore without outer intervention. Such approaches resemble passivity-based control strategies.

4.3.3.3 Reasoning about Organic Computing Systems

The existence of both environmental and operational uncertainty poses severe challenges when it comes to reasoning about the behavior of Organic Computing systems. In classically designed systems, testing is the main method being used to reach a certain level of confidence. In order to successfully deploy Organic Computing systems, we need to be able to derive at least an equal level of confidence about the performance of the system. As these systems are typically reconfigurable at run-time and often deliberate involve operational uncertainty, this necessarily involves stochastic reasoning. Such a stochastic reasoning can provide a number of benefits and can deliver strong guarantees. In traditionally designed systems, guarantees about a system's behavior often tend to become weaker as the system grows in size and complexity. In contrast, in systems consisting of probabilistically behaving and interacting elements, we are often able to extract stochastic guarantees about the system's aggregate behavior that tend to become stronger as the size of the systems grows. This closely resembles guarantees on bulk material properties we are quite used to in thermodynamics and material science. In this particular case, the decrease of uncertainty at the aggregate level is due to scaling effects and based on certain mathematical prerequisites like the enforcement of truly random and uncorrelated individual behavior. However, in many other cases it is less clear how the many small uncertainties related to a system's individual components can be composed to an aggregate picture.

In our discussion we also briefly highlighted potentially interesting further approaches that might be useful to make sound statements about Organic Computing systems, including reachability set analysis, stability notions in dynamical systems and control theory as well as verifiable probabilistic assumption guarantees (like e.g. the probabilistic model checker PRISM from the University of Oxford).

4.3.3.4 Conclusion and Challenges

Uncertainty is a concept of primary importance in the design of Organic Computing (OC) Systems. OC systems should be able to discover and modify models for environmental uncertainty and adapt and manipulate their own probabilistic behavior in accordance. For environmental uncertainty that is due to a lack of knowledge about the environmental conditions, it seems reasonable to employ approaches like e.g. self experimentation, learning and second-order probability techniques in order to proactively reduce uncertainty at least up to a reasonable level. In a sense, acknowledging the importance of uncertainty and probabilistic techniques in Organic Computing Systems retraces the findings of quantum mechanics which embraces probability theory and randomness as fundamental concepts that are necessary to adequately model our reality. Similarly, we now begin to understand the importance of uncertainty for the design and operation of complex technical systems. As Organic Computing community, we need to investigate what techniques from other disciplines can be used to handle the different kinds of uncertainty that are present in our technical systems. Furthermore, the Organic Computing perspective on complex engineered systems might be able to contribute new ideas and abstractions that can be used in other contexts. As a conclusion, it is justified to say that uncertainty is, at the same time, an important motivation, a tough challenge as well as a valuable tool in Organic Computing related research.

Participants

- Michael Beigl
KIT – Karlsruhe Institute of Technology, DE
- Jürgen Branke
University of Warwick, GB
- Uwe Brinkschulte
Univ. Frankfurt am Main, DE
- Sven A. Brueckner
Jacobs Technology Inc. – Ann Arbor, US
- Yuriy Brun
University of Washington – Seattle, US
- Jörg Hähner
Leibniz Univ. Hannover, DE
- Andreas Herkersdorf
TU München, DE
- Michael G. Hinchey
University of Limerick, IE
- Wolfgang Konen
FH Köln, DE
- Robert Laddaga
Doll Inc. – MA, US
- Chris Landauer
Aerospace Corp. – Los Angeles, US
- Falk Langhammer
Living Pages Research GmbH – München, DE
- Erik Maehle
Universität Lübeck, DE
- Christian Müller-Schloer
Leibniz Univ. Hannover, DE
- Phyllis R. Nelson
Cal Poly – Pomona, US
- Gabriele Peters
FernUniversität in Hagen, DE
- Marco Platzner
Universität Paderborn, DE
- Franz J. Rammig
C-LAB – Paderborn, DE
- Paul Robertson
Doll Inc. – MA, US
- Hartmut Schmeck
KIT - Karlsruhe Institute of Technology, DE
- Ingo Scholtes
Universität Trier, DE
- Bernhard Sick
Universität Kassel, DE
- Jürgen Teich
Univ. Erlangen-Nürnberg, DE
- Emil Vassev
University of Limerick, IE
- Laurenz Wiskott
Ruhr-Universität Bochum, DE
- Rolf P. Würtz
Ruhr-Universität Bochum, DE
- Johannes Zeppenfeld
TU München, DE
- Junmei Zhu
EBI – Cambridge, GB