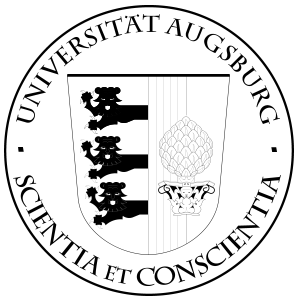
Universität Augsburg



Applications in Organic Computing

(Priority Program "Organic Computing" Nr. 1183 of the DFG)

Eds.: F. Nafz, M. Güdemann, W. Reif, H. Seebach

Report 2006-22 October 2006

Institut für Informatik
D-86135 Augsburg

Abstract

This technical report provides a summary of the applications and case studies used in the priority program "Organic Computing" of the German Research Foundation (DFG SPP 1183). All projects were asked to fill out a consistent form with several points to the respective application. These are presented in this report.

Contents

1 Introduction

Rising complexity and requirements claim new types of systems and other methods for development. These new systems should be intelligent, very flexible, robust, safe and adapt to human needs. They should control the complexity and could reach aims on their own without giving explicitly every command. Organic Computing based on the insight that we will have more and more autonomous systems, which have sensors and actuators to observe their environment and perform actions according to. An Organic Computing system is a system, which adapts dynamically to its environment and its conditions. Therefor, such a system will have several self-x properties, like self-organizing, self-healing, self-protecting, self-reconfiguring and context-awareness. There are a lot of different areas in which organic computing systems are conceivable. Therefore we want to present some example applications in the field of organic computing. In the following we present the case studies and applications used in the priority program "Organic Computing" of the German Research Foundation (DFG SPP 1183).

All projects filled out a consistent form about their project related application or the used case studies. The form has two parts. A first one in which the project, the application is related to, is described. It consists of the name of the project, a contact address and the location of the project. This is followed by a short description of the projects content. The second more detailed part is the application part. It consists of the application or case study and a listing of the self-x properties that appear. Further the applications are classified into several more general categories:

- Automotive and intelligent traffic
- Sensor networks applications
- Smart office
- Robot applications
- Chip design
- Biological systems and medical equipment
- Industrial image processing and computer vision

In the end of each form there are some references concerning the application and provide some more information for the reader.

2 Projects and Applications

2.1 AutoNomos

Name of project: AutoNomos: A Distributed and Self-Regulating Approach for Organizing a Large System of Mobile Objects

Contact: info@auto-nomos.de

Location: Braunschweig University of Technology
University of Lübeck

Short description of the project:

We propose AUTONOMOS, a distributed and self-regulated approach for the selforganization a large system of many self-driven, mobile objects. Based on methods for mobile ad-hoc networks using short-distance communication between vehicles, and ideas from distributed algorithms, local data clouds are formed in reaction to specific traffic structures (e.g., traffic jams). These Hovering Data Clouds (HDCs) are used for forming Organic Information Complexes (OICs) that are functional entities within the traffic flow, hosted by - but independent of - the individual moving vehicles (e.g., a structure detecting, indicating and monitoring the end of a traffic jam that continues to exist in place, even as the involved vehicles are replaced). Using HDC-based OICs, we develop Adaptable Distributed Strategies (ADSs) for dealing with complex and changing traffic situations. A final goal is the extension to achieve Global Objectives for vehicles and traffic flow. Our project is based on a wellestablished interdisciplinary cooperation and combines practical know-how from the field of mobile ad-hoc networks with theoretical expertise from a wide range of algorithmic topics.

Application:

Name: Traffic jam

Description:

A possible application is traffic jam recognition. In this scenario we consider single- or multilane roads. The formation and development of a traffic jam should be detected and information that affects other road users should be extracted. By inter-vehicle communication the front and back of a traffic jam is detected in the moment of its formation or shortly after. As a result, Hovering Data Clouds (HDCs) are established right at critical points, such as front and back of a traffic jam. As long as the underlying traffic jam exists, the HDCs should be maintained - independent of the participating vehicles. These data clouds are the base to construct an Organic Information Complex (OIC), an entity for high level traffic information that knows about the whole traffic jam structure.

Thus, we are not in need of local detectors, such as magnetic loops, and are able to transmit information immediately after the occurrence of the indicating event, without falling back on centralized registration and treatment and without the communication via traffic news on the radio.

Self-x properties of the application:

Self-organization: Road traffic is self-organized by all participating drivers, following their various destinations on different routes. To understand and describe such a self-organized structure, a system is needed that can handle those manifold degrees of freedom. All used techniques such as Hovering Data Clouds or Organic Information Complexes are selforganized to achieve high-level information without a central authority.

Self-adaption: Traffic structures are changing over time, so our algorithm has to recognize and react adequately to those changes. Additionally road users may change their behavior in reaction to our traffic information which in turn brings up another need of self-adaption. Self-healing: Communication in adhoc networks can be uncertain. Additionally vehicles can turn off anytime, what results in a changing infrastructure. HDCs are stable against those disturbances, since they are not addicted to a certain vehicle.

Self-optimization: In this traffic scenario, HDCs in combination with OICs have the purpose of optimizing traffic flows. By using self-organization techniques (in this case: creation of recommendations or directives for drivers), the system tries to minimize the necessary number of organic data structures - the better the traffic flows, the fewer HDCs and OICs are needed.

Classification: Automotive / Intelligent traffic

Application Reference:

[1] Wegener, Axel ; Schiller, Elad M. ; Hellbrück, Horst ; Fekete, Sándor P.; Fischer, Stefan: Hovering Data Clouds: A Decentralized and Self-Organizing Information System. In: Proceedings of International Workshop on Self-Organizing Systems, Passau, Germany, 2006

[2] Fekete, Sándor P. ; Schmidt, Christiane ; Wegener, Axel ; Fischer, Stefan: Hovering Data Clouds for Recognizing Traffic Jams. Submitted to 2nd International Symposium on Leveraging Applications of Formal Methods, Verification and Validation, 2006

2.2 ASoC

Name of project: ASoC: Architecture and Design Methodology for Autonomic System on Chip

Contact: A. Bernauer (bernauer@informatik.uni-tuebingen.de)
A. Bouajila (a.bouajila@tum.de)

Location: University of Tübingen
University of Munich

Short description of the project:

This project aims to develop an architecture and a design methodology for embedding autonomic or organic principles in System on Chip (SoC). We name this new kind of chip Autonomic System on Chip (ASoC). This project also proposes a design methodology for obtaining these systems. Future SoCs will witness a continued exponential increase in transistor capacity resulting in a complexity and reliability problem. We propose to rededicate a fraction of the abundant transistor capacity of future SoCs to implement organic computing properties. The system will have an increased fault tolerance, increased performance and power efficiency, easier system diagnosis and the capability to autonomously adapt to changing environmental conditions be it either externally imposed workloads or temperature variations. The project will only extend current SoCs to allow the reuse of existing IP.

Application:

Name: Design of an Autonomic System on Chip

Description:

An application is mapped to an architecture

Self-x properties of the application:

Self-organization: the configuration of the chip is partially determined during run time.

Self-optimization: choose the configuration that will trade between power and performance

Self-protect: try to predict approaching error situations and take necessary actions to avoid them (increased temperature, overload of processing units)

Self-healing: recover from erroronous states

Other special properties of the Application:

Graceful degradation: The system will try to perfrom the necessary tasks as

long as possible, even if the performance decreases.

Classification: Chip Design

Application Reference:

[1] Abdelmajid Bouajila, Johannes Zeppenfeld, Walter Stechele, Andreas Herkersdorf, Andreas Bernauer, Oliver Bringmann, and Wolfgang Rosenstiel. "Organic Computing at the System on Chip Level". In Proceedings of the IFIP International Conference on Very Large Scale Integration of System on Chip (VLSI-SoC 2006). Springer, October 2006.

2.3 The bio-chemical information processing metaphor as a programming paradigm for organic computing

Name of project: The bio-chemical information processing metaphor as a programming paradigm for organic computing
CHEMORG

Contact: Naoki Matsumaru (naoki@minet.uni-jena.de)

Location: Friedrich Schiller University Jena

Short description of the project:

All known life forms process information on a bio-molecular level. Examples are: signal processing in bacteria (e.g., chemotaxis), gene expression and morphogenesis, defense coordination and adaptation in the immune system, broadcasting information by the endocrine system, or finding a short route to a food source by an ant colony. This kind of information processing is known to be robust, self-organizing, adaptive, decentralized, asynchronous, fault-tolerant, and evolvable. Computation emerges out of an orchestrated interplay of many decentralized relatively simple components (molecules). This project will develop a theoretical and practical framework to exploit this bio-chemical information processing metaphor as a programming paradigm for organic computing. By doing so, we expect to make available a technology that allows to create computational systems with the properties of their biological counterpart. A couple of approaches are already using the chemical metaphor (e.g., Gamma, MGS, amorphous computing, and reaction-diffusion processors), but in accordance with Conrad's tradeoff principle, programming a chemical computer appears to be difficult. Therefore, we will focus - beside implementing a workbench for chemical computing - on developing and evaluating different techniques of "chemical programming". Furthermore, we will develop analysis methods based on our chemical organization theory. Finally, we will evaluate the new techniques quantitatively and compare them to conventional approaches. As a demonstrator application domain we aim at sensor networks, systems biology, and virtual actors.

Application:

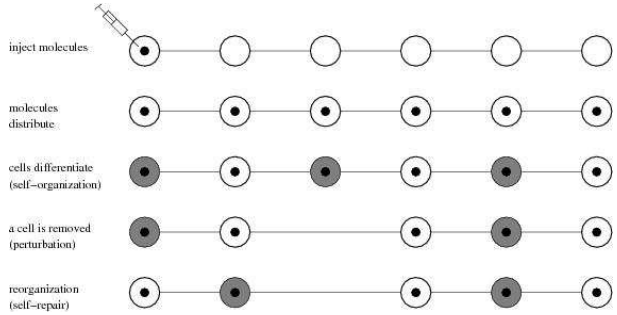
Name: Low-level differentiation control of distributed systems

Description:

As a possible application area for an organic computer implemented with bio-chemical information processing metaphor is wireless sensor networks. The sensor network consists of a huge number of small devices equipped with radio communication and sensors, and each sensor device is randomly located in space, spreading over a large area. It is impractical that a few main stations control such a large scale distributed system with even limited bandwidth. Thus,

self-organizing properties are wanted.

Furthermore, each entity in the network may interact with its environment via sensors and (possibly) actuators, so that it should be self-adaptive to environmental changes. A simple benchmark scenario is sketched as follows: Assume



that sensor nodes are arranged linearly and there are only two states for the nodes (grey or white). Since the computational components in biological organisms are biochemical molecules, the first step is to inject molecules that represent the chemical program. The injected molecules are distributed over the entire network. Then the network should self-organize such that pairwise neighboring nodes are in different states. For example, one class should perform a measurement at night the other at daytime. When nodes are removed or added dynamically, spontaneous reconfiguration should occur, controlled by the virtual chemical reactions among artificial molecules.

Differentiation of sensor nodes is a common mechanisms used for increasing the flexibility and performance of sensor networks. The adjacent sensor nodes are likely to cover the overlapping regions. Avoiding the redundant coverage and reducing the measurement resolution, the overall network lifetime can be extended. Reichenbach et al. [1] has suggested to assign a role as a cluster head to a sensor node to save battery energy. This assignment is an important illustration of the benefit of the node differentiation.

Self-x properties of the application:

Self-organization, self-reconfiguration and self-healing: The sensor node differentiation pattern is the result of an intrinsic self-organization process, without any global observer and controller. When one node is added to or removed from the network, the pattern is spontaneously reconfigured.

Classification: Biological systems

Application Reference:

[1] Frank Reichenbach, Andreas Bobek, Philipp Hagen, Dirk Timmermann. Increasing Lifetime of Wireless Sensor Networks with Energy-Aware Role- Chang-

ing. In Proceedings of the 2nd IEEE International Workshop on Self-Managed Networks, Systems & Services. 2006

[2] Naoki Matsumaru, Peter Dittrich. Organization-oriented Chemical Programming for the Organic Design of Distributed Computing Systems. In Bionetics 2006, Proceedings. 2006 (accepted)

2.4 Embedded Performance Analysis for Organic Computing

Name of project: Embedded Performance Analysis for Organic Computing

Contact: Steffen Stein (stein@ida.ing.tu-bs.de)

Location: Institute of Computer and Communication
Network Engineering (IDA), TU Braunschweig

Short description of the project:

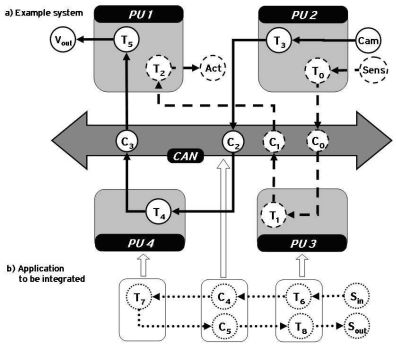
The aim of this project is to develop a framework for online performance analysis of heterogeneous embedded systems. We propose to concurrently run a distributed performance analysis framework on a given embedded system in order to compute run time performance data of the system. The framework should also be useable to evaluate different configuration scenarios of the embedded system and the applications running on it. This data can be used as input for algorithms performing adaptation or optimization of the running system.

Application:

Name: Online Acceptance Test and Optimization

Description:

The framework could for example be applied in an automotive scenario. Today's cars already have tens of networked control unites (ECUs) connected by bus systems. Integration of distributed applications running on the ECUs, communicating via shared buses is a hard problem the industry is facing today. Formal analysis approaches are being used to verify system timing behaviour. Integrating these approaches into the embedded system itself would enable updates and changes to these systems in the field without explicit engineering effort.



Take for example an embedded system and application setup as shown in figure

1. A system containing of four heterogeneous processing units (PU) connected by a CAN bus is running two applications (solid/ dashed chain). The application depicted by the solid chain gathers data from a camera controlled by PU 2, performs pre-processing on PU3, and post processing on PU 1. The application depicted by the dashed chain is a control task, reading data from a sensor, processing it on PU 3, to send a message to PU 1, which in turn controls an actor. Assume these applications have constrained end-to-end latencies and PU 3 as well as PU 4 is not fully loaded.

PU 3 and 4 not being fully loaded could motivate mapping a third end-to-end constrained application onto the system as depicted on the bottom of figure 1. Of course, integration of this application must not lead to violations of deadlines of the applications already mapped on the system. Also the new application must meet its deadline.

Online performance analysis implemented on the target system can integrate the new application to be mapped into its current model of the system and perform a what-if analysis of the resulting configuration. In case timing constraints are violated in the resulting setup, optimization algorithms can be run to find another suitable configuration of i.e. task mappings or priority assignments. In case none is found the update can be denied - the system gains self-protecting properties.

Similarly, if due to i.e. processor or bus degradation, the system setup changes, online performance analysis can check, whether the system can still comply with its real-time constraints and run optimization algorithms if needed. This adds self-healing properties to the system without the need of explicit redundancy.

Self-x properties of the application:

Self-awareness: The system has a notion whether it can comply with its real-time constraints. *Self-optimization:* The system can optimize itself to better comply with its real-time constraints or distribute load. *Self-protection:* The system can deny updates that would lead to violation of real-time properties. *Self-healing:* The system can adapt to subsystem failure or degradation by i.e. reassigning priorities or moving tasks. *Self-monitoring:* The system monitors itself to achieve self-awareness.

Other special properties of the Application:

Graceful degradation: If no suitable configuration can be found by optimization algorithms, irrelevant tasks can be removed from the system.

Classification: Automotive / Intelligent traffic

Application Reference:

[1] Steffen Stein, Arne Hamann, Rolf Ernst. Real-time Property Verification in Organic Computing Systems. In Proc. of 2nd International Symposium on Leveraging Applications of Formal Methods, Verification and Validation (ISoLA), Paphos, Cyprus, November 2006

2.5 Learning to look at humans

Name of project: Learning to look at Humans

Contact: Rolf Würtz (rolf.wuertz@neuroinformatik.rub.de)

Location: Ruhr-Universität Bochum, Institut für Neuroinformatik

Short description of the project:

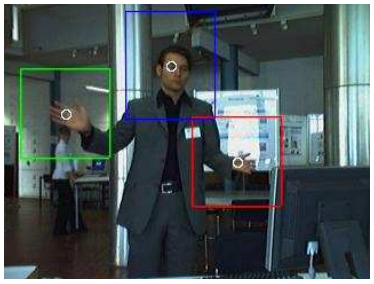
Artificial vision is a typical application domain of organic computing. Learning to interpret visual input is still a great challenge. We will develop a system that can learn to find and track humans in video sequences and to recognize individuals (as long as they don't change clothing). The system will be based on a generic data format specialized on the task only after learning; and on a general recognition mechanism (elastic graph matching). It will segment and extract models from video sequences based on a schematic definition of human gestures and fuse these individual models to gradually self-organize a generic articulated model. In the process, it will construct specialized subsystems for the recognition of body parts and will integrate these subsystems with each other. To recognize individuals, the system will form person-specific models emphasizing significant differences. While achieving the specific vision goals, the project will realize many of the important objectives of organic computing in an exemplary way.

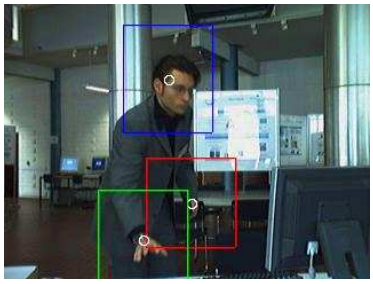
Application:

Name: Self-organized model building for person tracking

Description:

Module 1: Face and hand tracking
The goal of the project is a robust tracking system for moving persons. This requires the integration of several modalities and visual cues. Robustness can only be achieved if top-down models for the degrees of freedom of the human body are present in the application. Therefore, an important focus of is on learning the constraints that govern the articulated movement of the human body from example video sequences.





As part of a larger tracking utility we have realized a system that can track face and hands of a person standing frontally to the camera. It is robust enough to tell several dynamic gestures apart. It is based on a rather general system for the self-organization of a large number of visual agents. The importance of different visual cues is adapted automatically according to the agreement of its decision with other cues (democratic integration). This software module sets a couple of constraints on the relative position of conspicuous human features. In order to build a movement model it is important to track the limbs in between and identify them as separate entities, which can move relatively independent of each other.

Module 2: Color and motion based limb tracking
We have implemented a multi-agent system that is able to track the interior of lower and upper arm on the basis of movement and color cues. Here, a different kind of agents is used. Each agent is tracking its point on the basis of color and motion, and they are trying to fill a coherent region. Agents belonging to the same limb self-organize to a common subset. The relative movement of subsets will provide the basis for a statistical model of the degrees of freedom which a moving person has. In the images above, the lower arm is tracked reliably, while the upper arm area spreads out to the whole shirt, because it has uniform color and is not yet disambiguated by a different motion state (which the person has not performed yet). The technique is based on the algorithm by Lukas and Kanade, which is extended by a dynamical system, which destroys agents that have left a coherent region and creates new ones if a coherent region is not covered with sufficiently many agents. This module will be able to adapt to any kind of clothing.

Self-x properties of the application:

Self-organization: Agents interact in order to fulfill their goals, on a low level

there is no central control. In this way, the collection of all agents adapts to the changes in input data. On a lower level, the various tracking cues self-organize their relative weights, depending on environmental state.

Self-healing: The system is self-healing in the sense that agents that are destroyed due to changes in input data are recreated whenever more are required. In module 1 this leads to flexible behavior in the case of, e.g., self-occlusion. In module 2, the number of agents is dynamically regulated to fill the limb area as well as possible, which changes considerably during 3D-movement.

Other special properties of the Application:

Robustness: Due to the techniques of democratic integration and flexible creation and destruction of agents both modules work robustly with moderately deceptive backgrounds and in a wide range of environmental parameters such as lighting.

Classification: Industrial Image Processing and computer vision

2.6 MOVES

Name of project: Multi-Objective Intrinsic Evolution of Embedded Systems (MOVES)

Contact: Paul Kaufmann (paul.kaufmann@upb.de)
Marco Platzner(platzner@upb.de)

Location: University of Paderborn

Short description of the project:

Evolvable hardware denotes the combination of evolutionary algorithms with reconfigurable hardware technology to construct self-adaptive and self-optimizing hardware systems. The term evolvable hardware was coined by de Garis [1] and Higuchi [2] in 1993. Evolvable hardware is still a rather young area of research, and many issues and problems have not been addressed yet. Two main results have been achieved so far: First, evolutionary techniques are able to generate astonishing circuits that are totally different from classically engineered circuits, and sometimes even superior [3]. Second, for applications with time-varying specifications, very promising first results were achieved that indicate the potential of evolutionary techniques to construct self-adapting systems [4].

In this context our long term goal is the investigation and development of intrinsically evolvable embedded systems. Simulated evolution should provide embedded systems with a means to react properly to unforeseen changes in the environment and the system state. In an intrinsically evolved system, the evolutionary process runs together with the function under evolution on the same target platform. This is a necessary precondition for autonomous operation [5]. While our project is not focusing on a single concrete application, part of our work is the investigation of potential application domains. In the following, we present two applications where intrinsic evolution might be beneficial.

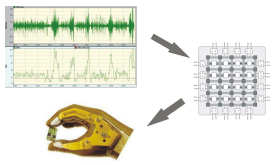
Application:

Name: : Prosthetic Hand Controller

Description:

Classification of the electromyographyc signals is a challenging task with many important applications. A prominent example is the control of a prosthetic hand based on measurements of an arm's EMG-signals with skin-attached sensors. The difficulty in doing so is that these signals, which are generated by the muscle activity, vary greatly in their spectrum and shape. This is for a number of reasons: Muscle fatigue lowers the frequency spectrum, skin conductance changes over the day and with age, the distance between muscles and skin varies, and signals from adjoining muscles interfere. The signals need a

substantial amount of preprocessing before they can be used to train or design a controller. Moreover, due to the changes in the signal characteristics the amputee has to re-train the controller periodically. The use of evolvable hardware for such an application has been proposed by Higuchi [6] and Torresen [7].



We consider creating a demonstration system with electromyographyc sensors on which we can investigate the short and long term adaptation capabilities of an intrinsic evolvable hardware controller. Such a demonstration system would allow us to gather realistic signal data to drive the design and evaluation of the evolutionary controller. Further, we could compare the evolutionary controller to other classification systems, such as neural networks.

Self-x properties of the application:

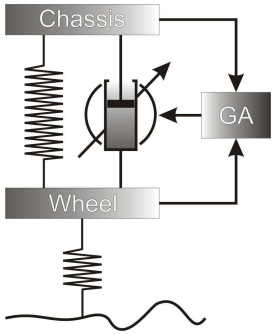
Self-adaptation is provided by the periodic re-evolution of the prosthetic hand controller. Self-optimization, organization and reconfiguration is inherently provided by the evolutionary algorithm running on a partially reconfigured hardware device.

Classification: Medical equipment — Controller design

Application:

Name: Semi-Active Suspension System

Description:



Comfort, speed, load carrying and road holding are the conflicting design criteria for a vehicle suspension system. Passive suspension systems are a predetermined compromise for the most likely operation conditions. Active or semi-active suspension systems allow the vehicle to react to varying conditions. Today's off-road and military vehicles, racing cars (active suspension systems are banned from formula one) and some luxury-type limousines are using active or semi-active suspension systems. To determine the current state of the system, acceleration sensors are being placed in the chassis and in the unsprung mass. From these measurements and the car body displacement, the control system derives other physical parameters such as velocities and jerks. Depending on the evaluation function applied, some of these parameters are combined into a single comfort metric. As an example, one of the most often cited metrics combines the cars body's jerk and the averaged position of the damper piston.

Recently, researchers studied the use of evolutionary techniques to design a semi-active suspension controller for a quarter-car model [8]. Our goal is to investigate the potential of an intrinsic evolvable hardware controller that would be able to adapt the suspension strategy to changing surfaces and load conditions.

Self-x properties of the application:

Self-adaptation is provided by i) the re-evolution of the suspension controller and by ii) switching to pre-evolved control strategies. Self-optimization, organization and reconfiguration is inherently provided by the evolutionary algorithm running on a partially reconfigured hardware device.

Classification: Automotive — Controller design

Application Reference:

[1] H. de Garis, Evolvable Hardware - Genetic Programming of a Darwin Machine, in Proceedings International Conference on Artificial Neural Networks and Genetic Algorithms (ICANNGA). Springer, 1993.

[2] T. Higuchi, T. Niwa, T. Tanaka, H. Iba, H. de Garis, and T. Furuya, Evolving Hardware with Genetic Learning: A First Step Towards Building a Darwin Machine, in Proceedings 2nd International Conference on Simulation of Adaptive Behavior (SAB). MIT Press, 1993, pp. 417-424.

[3] A. Thompson and P. Layzell, Analysis of Unconventional Evolved Electronics, Communications of the ACM, vol. 42, no. 4, pp. 71-79, 1999.

[4] T. Higuchi and N. Kajihara, Evolvable Hardware Chips for Industrial Applications, Communications of the ACM, vol. 42, no. 4, pp. 60-66, April 1999.

[5] P. Kaufmann and M. Platzner, Multi-objective Intrinsic Hardware Evolution, to appear in MAPLD 2006 International Conference.

[6] I. Kajitani, M. Murakawa, D. Nishikawa, H. Yokoi, N. Kajihara, M. Iwata, D. Keymeulen, H. Sakanashi and T. Higuchi, An Evolvable Hardware Chip for Prosthetic Hand Controller, Proc. of the Seventh International Conference on Microelectronics for Neural, Fuzzy, and Bio-Inspired Systems (MicroNeuro99), pp. 179-186, 1999.

[7] Jim Torresen. Two-Step Incremental Evolution of a Prosthetic Hand Controller Based on Digital Logic Gates, 4th International Conference on Evolvable Hardware (ICES2001), October 2001, Tokyo, Japan.

[8] A. Bourmistrova, I. Storey and A. Subic, Multiobjective Optimisation of Active and Semi-Active Suspension Systems with Application of Evolutionary Algorithm, MODSIM 2005 International Congress on Modelling and Simulation, December 2005, pp. 1217-1223

2.7 MODOC

Name of project: Model-Driven Development of Self-Organizing Control Applications (MODOC)

Contact: Helge Parzyjegla (parzyjegla@acm.org)

Location: Berlin University of Technology
University Stuttgart

Short description of the project:

Sensor- and actuator networks (SA-nets) will become an integral part of our living and work-ing environment. SA-nets consist of embedded controllers, mobile devices (e.g. PDAs, Smart-Phones), and sensors. These devices form complex wireless communication networks that are subject to unpredictable changes. Thus, it is impossible to completely predetermine the con-figuration at design-time, e.g. to decide which device executes which application components. Furthermore, faults and changes require constant reconfiguration. However, users do not want to administer their applications at run-time. Hence, applications have to be self-organizing in order to adapt to changing settings. For example, an application should reconfigure itself when faults occur or at least recover when faults have been removed. Self-organization re-quires knowledge about the application. It cannot be achieved by a middleware alone. Thus, the project proposes a model-driven development approach that encapsulates the required expert knowledge into the model transformation. Non-experts can develop applications using a high-level modeling language. A model transformation inspects the application model and synthesizes application components capable of self-organization and self-stabilization.

Application:

Name: MODOC for Smart Office / E-Home Scenarios

Description:

The proposed model-driven development process is applicable to different target domains, from which the smart office and e-home are very prominent example scenarios. Day-to-day devices such as PCs, PDAs, Smart-Phones, robots, TVs, HiFi-systems, and remote controls can provide control applications rendering our environments s̈martör ïntelligent:̈ To achieve this, the devices form a SA-net using networking technologies such as WLAN, Zig-Bee, Bluetooth, and IrDA. They communicate and cooperate with each other to anticipate the user's actions and to jointly offer the appropriate functionality and services within the de-tected context. Developing cooperating applications in this setting is a challenge and requires expert knowledge due to the heterogeneity of devices, the heterogeneity of networking tech-nologies, and also because of frequent configuration changes and common communication faults such as lost messages or

network partitions.

Today, many decisions have to be taken either by the developer at design-time or by the user at install- or run-time. But on the one hand, the developer can neither completely predeter-mine the configuration nor anticipate all potential faults that might occur at run-time. On the other hand, the user is also not willing or maybe not even able to handle complex configura-tion issues or faults at install- or run-time. He just wants to put the devices in place and the applications to run. Hence, SA-net applications must be self-organizing as well as self-stabilizing.

A major challenge is that the applications must deal with arbitrarily equipped SA-nets, be-cause every user may have a different set of devices and this set can even change over time when devices are added or removed. Therefore, the model transformation process inspects a given application model and identifies necessary tasks and their requirements (e.g., the avail-ability of a particular set of sensors or actuators) for a device to execute them. For each task a corresponding role is generated that a node fulfilling these requirements can adopt. We say an application is split into several cooperating roles that can be distributed on different nodes. Additionally, the model transformation equips the application with a self-stabilizing substrate that dynamically assigns roles to devices based on the capabilities a device has previously announced. Answering the question ẅho does whatät run-time increases flexibility and enables the application to organize itself adequately with respect to the particular SA-net it is deployed on.

Self-organization in our scenario comprises several aspects. First an initial configuration, i.e., a valid role assignment has to be found. Afterwards, roles must be monitored and reassigned if necessary. For instance, if a device performing an essential role is removed or fails, the ap-plication has to adapt and reconfigure itself by reassigning this role to another device. Fur-thermore, the opposite case of adding a device can also trigger a new role assignment that is caused by self-optimization in order to place related roles close to each other or to distribute the load a single nodes experiences. In general, the reassignment of a role will always succeed supposed that at least one node possessing the required capabilities exists. Otherwise, the role cannot be reassigned and the remaining roles of the particular application can also be turned off in order to save scare resources. Please note that other applications that do not depend on the removed or failed node can still continue working. Hence, the whole SA-net shows a graceful degradation by preserving as many applications as long as possible.

Transient faults such as lost or duplicated messages and arbitrary memory perturbations are addressed by keeping the generated roles and algorithms self-stabilizing. Just trying to mask these faults is not sufficient, since hardly all of them can be foreseen and appropriately masked by the developer. If an unmasked fault occurs, the application can get in an invalid state from which it may not recover without human intervention. But we cannot assume that the user is able to administer whole SA-net, since a crash can occur on any kind of appliance ranging from the coffee machine to the TV set. Instead self-stabilization guarantees that the application will recover from any transient fault after a bounded time (supposed that no other fault occurs meanwhile). However, this comes at a cost because the application can (in the general case)

not detect whether it currently is in a legal state or not. Self-stabilization only guarantees that an illegal state will be left after a fixed time by constantly pushing the system to valid state again. However, permanent faults such as failing nodes are subject to self-organization by reassigning roles as described above and restructuring the network accord-ingly.

Self-x properties of the application:

Self-organization: SA-nets as well as developed control applications must feature self-organization. Both have to adapt their structure in response to environmental changes, e.g., the addition or removal of devices. Roles have to be reassigned to nodes that are capable to take over the tasks of removed or failing nodes. Additionally the network has to reconfigure itself and adapt its routing paths in order to avoid network partitions. Therefore, several other self-x properties such as self-configuration, self-reconfiguration, self-monitoring, self-healing and self-optimization are required.

Self-configuration and self-reconfiguration: Initially a valid role assignment has to be found. In case of a failing node the affected applications have to reconfigure itself by reassigning the node's roles to other devices.

Self-monitoring: To detect a failing node and trigger a subsequent reconfiguration devices as well as assigned roles have to be monitored appropriately. Therefore, the self-stabilizing role assignment algorithm chooses one node to be responsible for this task and activates a special role on it for monitoring and coordinating the role assignment. Since every node is capable to act as a role coordinator no harm is done even if the current coordinating node fails-another node is chosen to take over. In summary, the SA-net monitors itself and ensures a valid role assignment.

Self-healing: In case of a failing node the affected roles are reassigned to other devices to keep the applications operational. We also refer to this as self-healing.

Self-optimization: A new role assignment can also be triggered by the detection of a newly added device. Roles can also be reassigned to better distribute the load a particular node ex-periences or to place related roles of one application close to each other in order to reduce communication costs.

Self-stabilization: Self-stabilizing algorithms and roles enable the applications to recover from invalid states caused by transient faults. Therefore, several techniques can be used ranging from fixpoint iterations over leasing-based approaches to precautionary resets. The model transformation process inspects the application model and equips the generated roles with an adequate mechanism with respect to the role's purpose and target domain.

Other special properties of the Application:

Graceful degradation: The role assignment algorithm tries to keep as many applications op-erational as possible. A particular application will run as long as for all of its roles at least one node can be found within the network that possesses the necessary capabilities of serving it.

Classification: Smart Office / E-Home

Application Reference:

[1] Torben Weis, Helge Parzyjegla, Michael A. Jaeger, and Gero Mühl. Self-organizing and Self-stabilizing Role Assignment in Sensor/Actuator Networks. In On the Move to Meaningful Internet Systems 2006: CoopIS, DOA, GADA and ODBASE, Proceedings, pages 1807-1824, Montpellier, France, November 2006. Springer.

[2] Mirko Knoll and Torben Weis. Optimizing Locality for Self-Organizing Context-based Systems. In Proceedings of the 1st International Workshop on Self-Organizing Systems (IWSOS 2006), pages 18-20, Passau, Germany, September 2006. Springer.

[3] Helge Parzyjegla, Gero Mühl, and Michael A. Jaeger. Reconfiguring Publish/Subscribe Overlay Topologies. In Proceedings of the 5th International Workshop on Distributed Event-based Systems (DEBS'06), Lisbon, Portugal, July 2006. IEEE Computer Society.

2.8 ORCA

Name of project: Organic Robotic Control Architecture - ORCA

Contact: Erik Maehle (maehle@iti.uni-luebeck.de)
Karl E. Großpietsch (karl-erwin.grosspietsch@ais.fraunhofer.de)
Werner Brockmann (brockman@informatik.uni-osnabrueck.de)

Location: University of Osnabrück, Institute of Computer Science
University of Lübeck, Institute of Computer Engineering
Fraunhofer Institute of Autonomous Intelligent Systems,
St. Augustin

Short description of the project:

The goal of the ORCA-project is to develop and evaluate an architecture and new methods based on organic computing principles for mobile autonomous robots in order to make them more reliable (concerning faults) and robust (concerning unforeseen situations) as well as eventually easier to design compared to classical (fault tolerance) approaches. In particular, general concepts inspired by the human autonomous nervous system and the human immune systems shall be applied. Thus, the robot shall be able to continuously monitor its own "health status" and ensure that it is stable and performing its task with optimum performance. Unusual, erroneous events and situations must be detected and handled in a way that the "survival" of the robot is assured without an explicit fault model and its mission is affected as little as possible. In contrast to more classical approaches, these unusual events and situations are not explicitly described in advance. Instead in case of a new and unknown deviation from the usual and healthy case, a counteraction is taken first, e.g. by selecting another behaviour or by adapting some system parameters. Based on its success or failure, the robot shall learn how to handle similar situations and to react then faster and more appropriate (as the human immune system learns how to fight reoccurring infections). This leads to a kind of emergent behaviour, i.e. the robot re-organizes its resources to adapt to the new situation in the best possible way. Thus it shall become self-healing, self-protecting, self-configuring, and self-optimizing. Though these organic principles are investigated for robotic scenarios, they should be general enough to be transferred to other complex embedded real-time systems, e.g. in industrial plants or future cars as well.

In particular the following requirements shall be met:

- An organic autonomous mobile robot should be capable of detecting irregular and unexpected sensor and behaviour patterns and react on them in a safe and goal-directed way.
- From reacting to unexpected patterns, emergent behaviour of the robot system should result. This kind of emergence requires to learn what to do and to remember how to react appropriately during the ongoing operation of the robot.

- Learning has to be done on-line and in-situ under hard real-time constraints. It must converge rapidly and must be stable against getting worse by further learning.
- Emergence should act flexibly, but within given boundaries in order to guarantee a safe behaviour of the robot and not to cause damage on the environment or the robot itself.
- Critical system states caused by learning counteractions must thus be avoided at any time.
- In order to assure stability and safety, the learning process must be controlled on-line such that no chaotic systems behaviour occurs due to totally free learning in a closed loop setting.
- Learning and the engineering of learning systems should be as intuitive and transparent as possible.
- A generic architecture shall be used for the organic computing system which can easily be engineered and reused.
- The required overhead should be as low as possible concerning implementation as well as engineering costs.

Application:

Name: Organic Self-Configuring and Adapting Robot - OSCAR

Description:

OSCAR is a six-legged ("Hexapod") autonomous walking robot. In total, OSCAR has 18 degrees of freedom (DOF), 3 DOF for each of its six legs. To sense its own status, OSCAR can measure the angle of each of these 18 joints as well as the current consumed by the controlling servo motors which is equivalent to the torque working on the joints. A sensor in each foot gives OSCAR tactile feedback: OSCAR can detect if its feet are on the ground or in the air. Complementing the proprioceptive sensors, OSCAR is equipped with ultrasonic range sensors to be able to perceive its environment. More sensors for proprioception and perception are to be installed in future.

The kinematics of OSCARs six legs has been biologically inspired. Stick insects also have six legs with a very similar geometric structure. Also the gait generation within OSCAR is inspired by stick insects. Instead of explicitly programming different gaits, only a few local rules determine the movement of each single leg (alternating swing and stance phases) and the coordination of all six of them (one leg may only swing if both neighbouring legs are on the ground). Depending on the desired walking speed, different gaits similar to those of stick insects evolve.

Multiple failures and defects can happen to an autonomous mobile robot like OSCAR. Acting in an unstructured and changing environment, a robot must be tolerant to situations where e.g. obstacles can not be detected very reliably.

Moreover the robot itself may introduce errors and failures or least problems induced by abrasion, lack of power, etc.

The redundant design of OSCAR allows it to tolerate some problems that may occur. We successfully simulated the failure of one or two legs. By changing only the neighbourhood relationship in the coordination rules for the legs, OSCAR is still able to walk after an amputation of one leg with only a small loss of performance. A second amputation can however only be tolerated if two opposing legs are affected.

Less radical defects of OSCAR's motors have already been simulated and Adaptive Filters have been successfully used to detect and compensate degradation and faults of the motors.

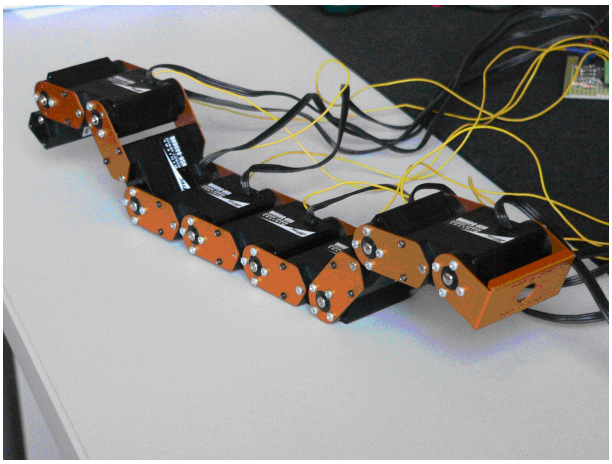
Application:

Name: Creeping Autonomous Robot for Learning demonstrations - CARL

Description:

Walking and climbing robots quickly get complicated due to their large number of degrees of freedom which makes the mechanical design as well as control complex and error-prone. On the other hand, wheeled mobile robots only have a limited need of coordination at lower control levels. Both limitations hamper basic investigations on emergent and self-organizing behaviour.

Therefore a second type of autonomous mobile robots was developed. It is as simple as possible, but yet robust and scaleable in order to concentrate on basic effects of emergent behaviour, learning and self-organization. Scalability is of importance here for testing the generalisability of methods and results. The family of the creeping robots CARL (Creeping Autonomous Robot for Learning demonstrations) hence is scaleable in two directions, namely the number of joints and the number of dimensions these joints may operate in.



The CARL robots move forward in a cyclic, wave-like manner. Similar to OSCAR, this is described only by interacting, local rules and yields an emergent, global system behaviour. Based on this motion, anomaly and fault-detection without an explicit fault-model are investigated. A signal, the so-called ´health-signal´, is generated as an indicator for the health-state of the robot in dynamic motion. It is solely based on measurements and a fuzzy-system to distinguish the unhealthy form the normal state of the robot. Also first steps were carried out to make the robot learn, i.e. CARL learned to optimize the step-width achieved by one cycle. Future learning addresses adaptation to the ground surface and to faults.

Self-x properties of the application:

Self-organization and self-reconfiguration: The gaits are not programmed explicitly but evolve in a self-organized fashion from a set of very simplistic rules. After an occurring failure of one leg, the coordinating rules for gait generation are changed to skip the affected leg. Following that structural change another gait evolves. As OSCAR can cope with the loss of a lateral leg better than a defect of a front or rear leg, the robot may reconfigure itself due to the symmetrical design of its body so that working legs are placed in the front and back of the robot and the faulty leg becomes a lateral one. Changing the assignment of the legs can change the walking direction of OSCAR in steps of 60 degrees.

Self-optimization: Using Adaptive Filters to compensate effects of degradation of the motors can lead to better performance in "normal" operation as well. Instead of manually optimizing motor controller parameters for one particular situation, these Adaptive Filters will self-optimize these parameters not only when failures occur. The same holds for the CARL robots when they adapt to the ground surface.

Self-healing: If a lateral leg brakes, the robot is still able to walk with only a little loss of performance. In the more problematic case of a loss of a front or rear leg, the reconfiguration of the robot leads to a better performance again. Although the robot will still not be fully functional, due to this self-healing

process an almost non-functional robot can recover to an operational albeit degraded state.

Self-monitoring: The robot control software of OSCAR is supplemented by "Organic Control Units" (OCUs) that continuously monitor other parts of the software and detect and react to changes of their behaviour. (Without analysing all possible errors and faulty situations that might occur, a "normal system state" is defined and/or learned by the OCUs. The ´health-signals´ then indicate the degree of deviation.)

Other special properties of the Application: *Graceful degradation:* Degradation of motor performance up to the loss of a whole leg can be tolerated and the robot will still be able to walk in situations where it would fail without self-organization and -reconfiguration of its legs and gaits.

Classification: Robot Application

Application Reference:

[1] Brockmann, W.; Maehle, E.; Mösch, F.: Organic Fault-Tolerant Control Architecture for Robotic Applications. 4th IARP/IEEE-RAS/EURON Workshop on Dependable Robots in Human Environments, Nagoya, Japan, Juni 16-18, 2005, Nagoya University/Japan 2005

[2] Großpietsch, K.-E.: Adaptive Filters for the Dependable Control of Autonomous Robot Systems. Proc. DPDNS 05 Workshop, Denver 2005

[3] Brockmann, W.; Großpietsch, K.-E.; Maehle, E.; Mösch, F.: ORCA - Eine Organic Computing-Architektur für Fehlertoleranz in autonomen mobilen Robotern. Mitteilungen der GI/ITG-Fachgruppe Fehlertolerierende Rechensysteme, Nr. 33, März 2006

[4] El Sayed Auf, A.; Mösch, F.; Litza, M.: How the Six-Legged Walking Machine OSCAR Handles Leg Amputation. Workshop on Bio-inspired Adaptive and Cooperative Behaviours in Robots, Rome, Sept 2006.

[5] Großpietsch, K.-E.; Silayeva, T.A.: Organic Computing - A New Paradigm for Achieving Self-Organized Dependable Behaviour in Complex IT Systems. Proc. IDIMT 2006 Workshop, Ceske Budejivice (Czech Republik), September 13-15, 2006

[6] Agarwal, R.: Intelligent Algorithms for Controlling an Autonomous Robot system. Master Thesis, Fraunhofer IAIS, Sept 2006

[7] Brockmann, W.; Großpietsch, K.-E.; Kleinlützum, K.; Maehle, E.; Meyer, D.M.; Mösch, F.: Concept for a Fault-Tolerant Control Architecture for CLAWAR Machines. 9. Int. Conf. Climbing and Walking Robots and the Support Technologies for Mobile Machines, Brussels, Belgium, Sept. 2006

[8] Mösch, F.; Litza, M.; El Sayed Auf, A.; Maehle, E.; Großpietsch, K.-E.; Brockmann, W.: ORCA - Towards an Organic Robotic Control. IWSOS - International Workshop on Self-Organizing Systems, Sept. 2006

[9] Jakimovski B.; Litza, M.; Mösch, F.; El Sayed Auf, A.: Development of an Organic Computing Architecture for Robot Control. GI-Workshop on Organic

Computing - Status and Outlook, Dresden, Oct. 2006

[10] Brockmann, W.; Meyer, D.M.; Horst, A.: Ein immunsystem-inspirierter Ansatz zum Überwachen des Lernens in Neuro-Fuzzy-Systemen. To appear: 16. Workshop "Computational Intelligence", Bommerholz, Nov. 2006

2.9 Organic Traffic Control

Name of project: Organic Traffic Control

Contact: Jürgen Branke (jbr@aifb.uni-karlsruhe.de)
Christian Müller-Schloer (cms@sra.uni-hannover.de)
Holger Prothmann (hpr@aifb.uni-karlsruhe.de)
Fabian Rochner (rochner@sra.uni-hannover.de)
Hartmut Schmeck (hsch@aifb.uni-karlsruhe.de)

Location: Universität Hannover
Universität Karlsruhe (TH)

Short description of the project:

The project Organic Traffic Control (OTC) focuses on the realization of a technically usable organic system, namely the decentralized control of road traffic. Within the project, an archi-tecture for learning traffic light controllers is being developed. Such a learning traffic light controller quickly reacts to changing traffic demands at the controlled node by adapting the timing of the traffic lights. A learning capability allows an autonomous reaction to unforeseen traffic situations. A network of these adaptive learning traffic light controllers is being im-plemented in a simulated city environment to study the possibilities and limitations of decen-tralized (traffic) control systems.

Application:

Name: Decentralized Traffic Control

Description:

The control of traffic lights in urban road traffic networks is a challenging task. Traffic flows in these networks are changing constantly and on different time scales. There are small short-time variations around a constant mean arrival rate and there are large intra-day changes that occur e.g. due to commuter traffic. Unfortunately, not every change in traffic can be foreseen since flows might change due to public events, road works, or incidents in the vicinity of a node. Therefore, a mere generation of a single "optimal" traffic light controller for a certain traffic demand is not sufficient. Sophisticated traffic light controllers need the ability to adjust quickly to changes in traffic situations and to react reasonably in situations that have not been anticipated in their design process.

Existing traffic controllers can be very broadly divided into two approaches:

- Fixed-time traffic controllers switch their traffic lights following a predefined sched-ule. They are not traffic-responsive, but different schedules can be used depending on the time of day to approximately accommodate intra-day changes. The advantage of this approach is that several traffic

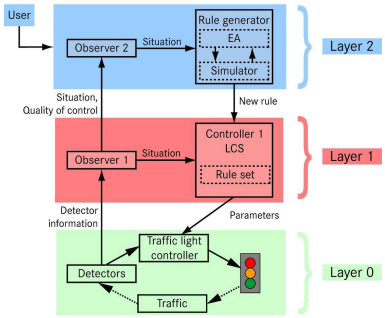


Figure 1: Architecture overview. Layer 0 represents the traffic node, Layers 1 and 2 are organic control layers responsible for the selection and generation of signal programs

nodes can be coordinated quite easily to form syn-chronized traffic lights (green waves).

- Traffic-responsive strategies utilize traffic data provided by detectors to adapt their phase durations to short-term variations. The exact behaviour of a controller is deter-mined by a number of parameters that often need to be adapted to accommodate intra-day changes adequately. Due to their reactive freedom, traffic-responsive controllers are hard to coordinate.

Research in the OTC project focuses on the development of an adaptive traffic light controller architecture with learning capabilities. The overall architecture is traffic-responsive and can adapt to larger changes in traffic due to a simulation-based learningapproach. Therefore, it is supposed to be self-optimizing. Its three layers are depicted in Figure 1.

On Layer 0 a conventional traffic light controller is used to control the traffic lights. The con-troller can be parameterized. It might implement a fixed time scheme, or it can be traffic-responsive. In the first case, the phase sequence and durations can be specified as parameters, in the second case, the parameters specify different aspects of the control process (like e.g. the minimum and maximum duration of a phase and when to extend a phase over its minimum duration).

Layer 1 monitors the traffic situation at the node and adjusts the parameters of the traffic light controller when necessary. This is done using an observer component aggregating traffic data collected by the detectors on Layer 0. This data is used to compute a set of flow values that characterize the traffic situation. The detected situation serves as input for a modified Learn-ing Classifier System (LCS) which selects appropriate parameters for the traffic light control-ler on Layer 0. Parameter selection is based on the past performance of the parameters in similar situations. This LCS serves as the controller on Layer 1.

Whenever a situation is not adequately covered in the LCS, appropriate parameters (describ-ing a traffic light controller) for this situation are generated by simulation-based optimization on Layer 2. An evolutionary algorithm (EA)

generates populations of parameters and evalu-ates them using a microscopic simulation model. At the end of this process the generated pa-rameter set becomes available to the LCS.

Within the OTC project, the proposed controller architecture is being implemented in a simu-lated urban environment. Further extensions of the architecture are planned to be able to cope with larger and more complex traffic networks. This is a precondition for further research on the influence and usability of controlled self-organization on/for traffic control.

Self-x properties of the application:

Self-adaptation and self-optimization: The architecture provides means for adaptation to changing traffic situations by monitoring the traffic and selecting appropriate parameter sets. New optimized parameter sets will be generated whenever this is necessary.

Other special properties of the Application:

Simulation-based learning: The proposed architecture contains a simulation model of the traf-fic network to create new parameters for previously unforeseen situations. By simulating the network it is possible to evaluate a large number of parameter sets in a short period of time. "Bad" parameter sets can be detected through simulation and will not be applied in the real network.

Classification: Automotive / Intelligent traffic

Application Reference:

[1] Fabian Rochner, Holger Prothmann, Jürgen Branke, Christian Müller-Schloer, and Hartmut Schmeck. An organic architecture for traffic light controllers. In Christian Hochberger and Rüdiger Liskowsky, editors, Informatik 2006 - Informatik für Menschen, volume P-93 of LNI, pages 120-127. Köllen Verlag, 2006.

[2] Jürgen Branke, Moez Mnif, Christian Müller-Schloer, Holger Prothmann, Ur-ban Richter, Fabian Rochner, and Hartmut Schmeck. Organic Computing - Addressing complexity by controlled self-organization. In Proceedings of the 2nd International Symposium on Leveraging Applications of Formal Methods, Verification and Validation (ISoLA 2006). (To appear.)

2.10 Quantitative Emergence

Name of project: Quantitative Emergence: Metrics, Observation and Control Tools for Complex Organic Ensembles - QE

Contact: Urban Richter (uri@aifb.uka.de)
Moez Mnif (mnif@sra.uni-hannover.de)

Location: Universität Karlsruhe, Institute of Applied Informatics and Formal Description Methods (Institute AIFB)
Universität Hannover, Institute of Systems Engineering, Dep. System and Computer Architecture (SRA)

Short description of the project:

This joint project with the Universität Hannover and the Universität Karlsruhe (TH) will focus on the design of concepts and tools for the implementation of an architecture needed for the realisation of self-organising technical systems which are, at the same time, reliable, robust, and adaptive. As an important prerequisite for designing such systems we have to understand the effects of emergent global behaviour in networks of intelligent autonomous units. Hence, we have to quantify emergent behaviour and we need tools to prevent unwanted behaviour and to encourage or enforce desired positive effects. The architecture will consist of a network of autonomous units (called "system under observation and control" (SuOC)), one or more observers, and one or more controllers. We shall develop an appropriate methodology for observing the (global) behaviour of the system and for quantifying and evaluating effects of emergence (this will be done mainly at Hannover). Furthermore, we have to generate adequate responses of the controller to the results of the observer in order to enable a controlled emergent behaviour within the restrictions set by some external unit (the environment). This requires an exploration of various potential ways of influencing the behaviour of a self-organising production system. (This will be done at Karlsruhe). For the initial 2 years of this project we shall abstract from realistic technical applications and will use rather simple artificial production systems exhibiting some interesting properties. Only later, during Phase II (i.e. years 3 and 4) we intend to combine the new concepts and tools for the observer and the controller with the more complex organic production system Organic Traffic Control which will be developed in the corresponding project OTC.

Application:

Name: Elevator group control

Description: Elevator group control is a familiar problem to everyone, who has used an elevator system in a high office building or skyscraper, and its conceptual simplicity hides significant difficulties. Operating in continuous state spaces, non-stationary behaviour of elevators and passengers, or changing pas-

senger arrival rates are just some indicators that we should mention. In the last decades, elevator group control has been studied as a well-known example within different research domains, but an optimal and globally valid policy for group control is not known and depends on many determining factors.

Elevator group control serves as our test bed to develop and evaluate a generic architecture for Organic Computing systems. We define an artificial, decentralised scenario, where every elevator is responsible for its own. An elevator doesn't know what the other elevators are doing. It just sees the hall calls, which are driven by passenger arrivals, moves up and down, while capacity is not empty and passengers have entered the cabin, follows a simple moving strategy, and tries to minimise the global average waiting time of passengers. Different arrival patterns during the course of the day need different strategies, but in the first step of our research we concentrate on inter-floor traffic. Up-peak traffic in the morning rush hour of a typical office building, where every elevator is filled up in the lobby with passengers and stops at many different upper floors, or down-peak traffic, which shows a reversed situation, many arrival floors and one destination, are ignored.

Our test bed implementation follows the oldest known principle of collective control. Every cabin always stops at the nearest hall call in its running direction. One major disadvantage of this strategy is that the emergent phenomenon called bunching effect is not avoided. Several elevators may synchronise, arrive at one floor at the same time, serve the next hall calls together, move synchronously over a longer period of time, and might make the average waiting time of the passengers much longer. This behaviour, which is equal to a parallel wave that moves up and down from floor to floor, is quite inefficient, because elevators, participants of the bunching effect, can be substituted with one huge single elevator with a capacity equal to the sum of the individual synchronised cabins. This huge single cabin looses the advantage of flexibility and the possibility to serve hall calls on different floors at the same time. But we can observe some other type of lower bunching. The elevators are synchronised, move up or down in the same direction over a longer period of time, but have a fixed gap of some floors between them. In this case other control strategies are needed to dispatch the cabins, and a huge cabin will not solve the problem of inefficiency. From the point of view of Organic Computing the bunching effect shows an interesting emergent behaviour in a technical scenario. The system under observation and control operates inefficiently, the average waiting time increases, and the system performance decreases. An elevator group, which uses collective control and doesn't have any interaction between the cabins, e.g. the elevators have information about the direction or the position of the other elevators, produces bunching, which is an emergent behaviour that can be measured by a global observer and controlled by an organic controller.

We use this test scenario to construct metrics for observation, to develop control strategies, and to evaluate a generic observer/controller-architecture for Organic Computing scenarios to cope with such unwanted, and emergent behaviour or failures in complex systems. Our objective is to contribute elements of a toolbox with basic mechanisms to observe, analyse, and control emergent behaviour, which can be used in other scenarios for Organic Computing.

Self-x properties of the application: The decentralised elevator group control illustrates a preliminary test scenario in our project. In the first instance we develop a generic architecture for Organic Computing systems, we develop and investigate metrics for the observation and analysis of emergence in self-organising systems and we develop and investigate mechanisms to influence and control the effects of emergence in these systems. The elevator scenario is used to validate our tools, i.e. our objective is not to develop a new and a better strategy for elevator group control, which is competitive to existing solutions. Our test scenario exhibits several different examples of self-x properties, which are interesting in the context of Organic Computing.

Self-monitoring: The observer monitors the behaviour of the elevator group, aggregates system data and observes different types of bunching of the elevators. As we mentioned above the bunching effect can be characterised as a parallel wave or as a group of cabins, which move in the same direction over a longer period of time and have a fixed gap between them. Bunching is classified as an error, and the described effect of a parallel wave results in a less efficient performance of the elevator system. The global goal will be to minimise system errors, while the observer recognises and predicts emergent behaviour earlier.

Self-organisation: The controller gets results of data measurement and analysis from the observer and influences the elevator group in order to prevent bunching. The system controls itself to reduce waiting times as a global goal and it acts self-organised with predefined degrees of freedom. The controller defines the best control action for an observed situation, tries to dispatch the cabins, or prevents the bunching effect, and learns over time, which control strategy works as best in a special situation.

Self-optimisation: The controller can choose between different control strategies to achieve the system optimum, or just to reduce the waiting time. Selection of control strategies is done with a learning classifier system, which self-optimises itself by genetic operators that mutate and recombine control actions.

Self-adaptation: The learning classifier system is a mechanism of machine learning. It learns new measured situations, and finds best control actions for this situation.

Self-healing: If bunching occurs the system has the ability to measure and control this emergent behaviour on its own. The elevator group serves all hall calls with minimised waiting times, and the system won't break down.

Classification: Automotive / Intelligent traffic; Robot application

Application Reference:

[1] Branke, J., Mnif, M., Müller-Schloer, C., Prothmann, H., Richter, U., Rochner, F., and Schmeck, H.: Organic computing - addressing complexity by controlled self-organization. To appear in Proceedings of ISOLA 2006.

[2] Richter, U., Mnif, M., Branke, J., Müller-Schloer, C., Schmeck, H.: Towards a generic observer/controller architecture for organic computing. In Hochberger, C., Liskowsky, R., eds.: INFORMATIK 2006 - Informatik für Menschen! Vol. P-93, GI-Edition - LNI, Bonner Köllen Verlag (2006) 112-119

2.11 Quantitative Emergence

Name of project: Quantitative Emergence: Metrics, Observation and Control Tools for Complex Organic Ensembles - QE

Contact: Urban Richter (uri@aifb.uka.de)
Moez Mnif (mnif@sra.uni-hannover.de)

Location: Universität Karlsruhe, Institute of Applied Informatics and Formal Description Methods (Institute AIFB)
Universität Hannover, Institute of Systems Engineering, Dep. System and Computer Architecture (SRA)

Short description of the project: This joint project with the Universität Hannover and the Universität Karlsruhe (TH) will focus on the design of concepts and tools for the implementation of an architecture needed for the realisation of self-organising technical systems which are, at the same time, reliable, robust, and adaptive. As an important prerequisite for designing such systems we have to understand the effects of emergent global behaviour in networks of intelligent autonomous units. Hence, we have to quantify emergent behaviour and we need tools to prevent unwanted behaviour and to encourage or enforce desired positive effects. The architecture will consist of a network of autonomous units (called "system under observation and control" (SuOC)), one or more observers, and one or more controllers. We shall develop an appropriate methodology for observing the (global) behaviour of the system and for quantifying and evaluating effects of emergence (this will be done mainly at Hannover). Furthermore, we have to generate adequate responses of the controller to the results of the observer in order to enable a controlled emergent behaviour within the restrictions set by some external unit (the environment). This requires an exploration of various potential ways of influencing the behaviour of a self-organising production system. (This will be done at Karlsruhe). For the initial 2 years of this project we shall abstract from realistic technical applications and will use rather simple artificial production systems exhibiting some interesting properties. Only later, during Phase II (i.e. years 3 and 4) we intend to combine the new concepts and tools for the observer and the controller with the more complex organic production system Organic Traffic Control which will be developed in the corresponding project OTC.

Application:

Name: Control of emergent behaviour of distributed agents by the example of a chicken simulation

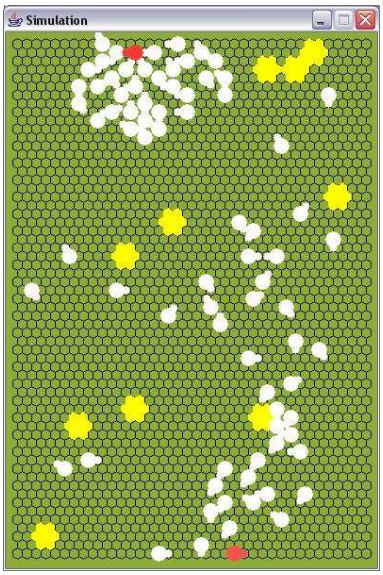
Description:

The objective of our work is to make emergent phenomena accessible to a practical measurement and control process. This is important in self-organising technical applications that need the possibility of detecting emergent phenom-

ena in order to support or to suppress them.

The application described below (chicken simulation) is inspired by nature, and shows clustering from a global point of view as an emergent behaviour of local interaction. The goal of this application is to get a better understanding of the global emergent behaviour of distributed agents. It serves also as test bed to validate our generic observer/controller architecture, which is elaborated in [1].

The chicken simulation [2] reproduces the collective cannibalistic behaviour of densely packed chickens in cages (cooperation with the University of Veterinary Medicine Hannover), and tries to explain this unwanted behaviour, which is frequently observed when a chicken is injured, which leads to a major loss of animals (up to 50% of the animals). If chickens perceive a wounded chicken, they chase and pick on it. This leads to the emergent building of chicken swarms (or clusters). A swarm disperses when the wounded chicken is killed. The emergent behaviour in this scenario is spatial, but swarms move over time. This is a case of "negative", i.e. unwanted, emergence, since the global goal should be to reduce chicken death rate.



While simulating this behaviour, order patterns emerge as expected in form of chicken swarms. At the moment these patterns are interpreted by human experts. Our goal from the point of view of Organic Computing is to classify them automatically. To achieve the goal of maximising the lifetime of chickens, we introduce the observer / controller paradigm. The observer reports a quantified context of the underlying system to the controller. The controller evaluates the situation and reacts with adequate control actions to disperse swarms or to prevent their formation.
Organic Computing focuses on the problem of increasing complexity in technical scenarios. The chicken simulation has no obvious technical approach on its own. However it can be used to study general multi-agent systems bearing obvious similarities to technical systems. In the simulation a chicken is directed by predefined rules, and will be influenced by the behaviour of other chickens in its local neighbourhood or by the environment, e. g. by noise that frightens them and feed that attracts them. Therefore, chickens are considered as autonomous robots or agents with simple rules und local goals. There are many analogue technical scenarios with a similar structure, as described in [3]. This analogy justifies our biological inspired simulation and shows a technical relevance for Organic Computing.

Self-x properties of the application: *Self-monitoring:* The observer monitors the behaviour of the chickens, and aggregates system data by quantifying the underlying situation. The formation of chicken swarms is considered as

a negative emergent effect, that affects the global goal of the system (reduce chicken death rate).

Self-healing: The system adapts to failure situations like the injury of a chicken. It has the ability to measure and to prevent the resulting emergent phenomenon of clustering.

Self-organisation: The chickens organise themselves to swarms. Every time this phenomenon occurs, the controller interferes by changing the environment (e.g. noise or feed) to get a better fulfilment of the global objective. The chickens react by dispersing over the area.

Classification: Robot application

Application Reference:

[1] Urban Richter, Moez Mnif, Jürgen Branke, Christian Müller-Schloer and Hartmut Schmeck. Towards a generic observer/controller architecture for Organic Computing. In INFORMATIK 2006 - Informatik für Menschen! Bonner Köllen Verlag, 2006.

[2] Mnif, M., Müller-Schloer, C.: Quantitative emergence. In: Proceedings of the 2006 IEEE Mountain Workshop on Adaptive and Learning Systems (IEEE SMCals 2006). (2006) 78-84

[3] Markus Jäger. Kooperierende Roboter: Gemeinsame Erledigung einer Reinigungsaufgabe. KI, 18(2):59-, 2004.

2.12 SAVE ORCA

Name of project: Formal Modeling, Safety Analysis and Verification of Organic Computing Applications - SAVE ORCA

Contact: Wolfgang Reif (reif@informatik.uni-augsburg.de)
Florian Nafz (nafz@informatik.uni-augsburg.de)

Location: University of Augsburg

Short description of the project:

The subject of this project is a method for the systematic, top-down design and construction of highly reliable and adaptive organic computing applications. Reliability here means preservation of functional correctness, safety and security under unexpected disturbances and component failures. Adaptability in the context of this project is related to adaptive system behaviour under changing requirements and modified tasks. The aim is to provide a formal framework for descriptive requirements, rigorous modeling and modular design of self-adapting and self-organizing systems. This will make design and construction of future organic systems easier and safer. As an integral part of the framework, formal analysis will improve reliability, stability, and adaptability of such systems. The approach combines formal specification and verification with failure analysis techniques and intelligent re-configuration. Example applications are 'automation engineering' and 'mobile systems'. A formal framework for the analysis of organic computing applications will be provided and prototypical tool support will be implemented. The method will be evaluated with substantial case studies.

Application:

Name: Self-adaptive robot cell

Description:

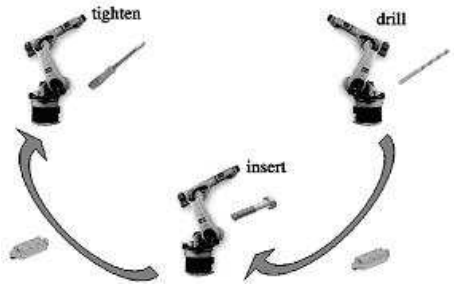
One application is an automated production cell with three robots, which are connected with autonomous transportation units. Every robot can accomplish three tasks: drilling a hole in a workpiece, inserting a screw into this hole and tightening the screw. These tasks are accomplished with three different tools that can be switched. Every workpiece must be processed by all three tools in the given order (drill, insert and tighten = DIT). Workpieces are transported between the robots by autonomous transportation units (carts). Changing the tool of a robot requires some time. Therefore the standard configuration of the system is that the three tasks are spread out between the three robots and the carts transfer workpiece accordingly.

We examine the case when one or more tools break and the current configuration allows no more correct DIT processing of the incoming workpieces. If

the driller of one robot breaks and DIT processing is not possible, as no other robot is configured to drill. A non-adaptive production cell would now come to a standstill and wait for maintenance.

However, it is obvious, that this is not a real hazard as the robots have three tools and can switch to another tool if one breaks. So it should be possible for the adaptive system to detect this situation and reconfigure itself in such a way, that DIT processing becomes possible again. This implies that at least one other robot also has to switch its tool, so that all three tools are available again.

This can be resolved by reassigning the tasks and switching the tools. Now one robot drills, another robot tightens the screws and the third robot is left unchanged. For this error resolution, not only the assignment of the tasks to the robots must be changed, but also the routes of the carts. If only the tools were switched, the processing of all tasks would be possible, but not in the correct order.



Another form of adaption is possible in the production cell when partially processed workpieces arrive. We could for example put a RFID tag to the workpieces that indicates whether they have already a task finished, for example a drilled hole. Those that have the hole drilled don't have to be processed by the robot assigned the drill task. If there is an additional transport cart available, then it can bring the partially processed workpieces directly to the robot that inserts the screws. This is an example of self-adaption by appropriate reaction to changing tasks.

When this has happened, the robot that has been assigned to the task to insert the screws might become the bottleneck in the system. If an additional robot and cart are available, they can be integrated to optimize the throughput of the production cell.

Again, self-x properties are only possible because of internal redundancy in the system. This includes redundant tools that can be switched and redundant robots or transport carts. The difference to traditional redundant systems is that they are dynamically configured and may be used for other tasks if not needed at the moment.

Another form of self-adaption that can be illustrated by our example is graceful degradation. This means that the system tries to fulfill at least parts of its functions as long as possible. In this example this could be drilling holes in a

workpiece and inserting screws but not tighten them. Yet another form would be to use one robot to accomplish more than one task. This also falls under graceful degradation because it preserves the functionality but diminishes the throughput of the production cell considerably (under the assumption that switching tools takes quite a while).

In summary the adaptive systems tries to preserve its functionality as long as possible even under failure of tools. This is achieved by self-configuration and works as long as there is enough internal redundancy in the system and processing is possible.

Self-x properties of the application:

The production cell example exhibits several different examples of self-x properties.

Self-adaption: Adaption to changing goals like different or already partially processed work-pieces is achieved by assigning different routes to the autonomous carts that deliver the workpieces to the corresponding robots.

Self-organization and self-reconfiguration: After an occurring failure the production cell assigns the new tasks in a correct way to the robots, so that production can continue. The robot reconfigure themselves according to the assigned task.

Self-optimization: If an additional robot and cart is available, they can be integrated to optimize the throughput of the production cell. This should be done by the Systems itself.

Self-healing: If a tool brakes the production cell compensates this by reassigning the tasks and the robots change the tools accordingly.

Self-monitoring: The productions cell recognizes an error, like DIT processing is no more possible, on its own. The Robots observe there tools and notice if a tool is broken.

Other special properties of the Application:

Graceful degradation: If the system is not able to reconfigure the cell by reassigning the tasks after a tool broke, it tries to fulfill at least parts of its functions as long as possible, but with limitation, for example reduced throughput. A robot could tighten some workpieces, then change the tool and insert a screw, for example.

Classification: Robot application

Application Reference:

[1] Matthias Güdemann, Frank Ortmeier and Wolfgang Reif. Formal Modeling and Verification of Systems with Self-x properties. In Autonomic and Trusted Computing 2006, Proceedings. Springer LNCS, 2006

2.13 Energy Aware Self Organized Communication in Complex Networks

Name of project: Energy Aware Self Organized Communication in Complex Networks

Contact: Jakob Salzmann (Jakob.Salzmann@uni-rostock.de)
Dirk Timmermann (Dirk.Timmermann@uni-rostock.de)

Location: University of Rostock

Short description of the project:

Networks surrounding us are growing and getting even more complex because of actual progresses in ubiquitous and pervasive computing and distributed wireless networking. Conventional methods and mechanisms are not suitable any more. Possible solutions can be found in nature which offers mechanisms and techniques resulting in enormous benefits with just little effort. Self organization of complex systems and the arising of complex emergent global behaviour from simple local rules are the most promising and still mysterious areas to explore. Nature works following the principle "Simple rules rule.", although this is not always obvious. The main task is to discover the simple rules and mechanisms behind the complex appearing patterns of behaviour and natural processes to make them applicable on a technological level. On the one hand, we still have less knowledge on these principles and a lot of research work is needed. We have to investigate what natural phenomena can be adapted, how they function and what advantages can be expected. On the other hand, we have to unbind ourselves from conventional engineering methods and to step on novel ways. In the context of this application, we target on exploring the interrelations between causes and effects in complex and scale free systems. Sensor networks will be used as paradigm for the biological structures and natural phenomena we want to examine. Emerging appearances like stigmergy, altruism and graceful degradation will be regarded. Our aim is the emergence of energy aware communication methods in complex networks by designing and altering local rules.

Application:

Name: Forest fire surveillance

Description:

One application for a network of sensor nodes is the surveillance of a forest fire. The network consists of several hundred nodes which were dropped by an air plane, for example. It is assumed that these nodes are distributed randomly on the area. Each node is equipped with a transmitter and a receiver, a temperature sensor and a battery with limited capacity.

The task of the network is to detect and track a possible forest fire by measuring the temperature and transmit the measured data to a central station, called sink. The effort for communication and measurement tasks has to be reduced to a minimum at each node to save energy and to maximize the network's lifetime. Another task of the network is to handle situations involving defective nodes to increase robustness and reliability.

The potentially high distances between sink and node ask for a multihop communication strategy. For large numbers of nodes a direct control of each node by the sink would generate much traffic along a multihop route. A self organization strategy is expected to reduce the needed communication between sink and nodes to a minimum.

Depending on their position in the network, their distance to their neighbours and their remaining energy, nodes decide theirself to join or to lead a cluster of nodes. The task of the Clusterhead is to enable routing to adjacent clusters, to the sink or to assign nodes as gateways. Subject to the remaining energy of one cluster and the estimated risk of a forest fire, a clusterhead autonomously decides to shutdown or to activate redundant nodes in its cluster. In areas with low temperature, a Clusterhead will reduce the number of active nodes and the rate of their measurements to a minimum. In regions with high temperature, a Clusterhead will activate all remaining nodes to localize the position of the fire most exactly. If for any reason the Clusterhead failes, the remaining nodes of the cluster have to determine a new leader and resume work.

The ability of each node to become Clusterhead, gateway or a standard node is one of the necessary redundancies to gain self-x properties. Additionally the random distribution of the nodes causes some areas to have more nodes than others. With the concept of stigmergy these redundancies can be used to optimize routing and clustering.

Self-x properties of the application:

Self - Organization: To reduce communication traffic, the nodes of the network divide their selves into clusters. They decide to join or to lead a cluster, depending on their position and remaining enegry.

Self - Healing: After a possible failure of a Clusterhead, the remaining nodes of a cluster have to determine a new Clusterhead and resume work.

Self - Adaption: Depending on the risk of a fire and their remaining energy, the clusters change their fire detection activity autonomously.

Other special properties of the Application: Graceful degradation: If the remaining energy of the nodes is too low, Clusterheads can decide to switch off such nodes to save energy by reducing communication traffic. But, the accuracy and the ability to detect fire alsodecreases.

Stigmergy: Depending on the number of their adjacent nodes, their remaining energy, their cluster and their position, nodes will change their own probability to become Clusterhead or gateway. A routing path will be established differently according to the enviroment.

Classification: Sensor network application

Application Reference:

[1] Reichenbach, F.; Bobek, A.; Hagen, P.; Timmermann, D. Increasing Lifetime of Wireless Sensor Networks with Energy-Aware Role-Changing, In Proceedings of the 2nd IEEE International Workshop on Self-Managed Networks, Systems & Services (SelfMan 2006), Dublin, Ireland, Juni 2006

2.14 Smart Pixels

Name of project: Organic architectures for self-organising smart pixel sensor chips

Contact: Dietmar Fey (dietmar.fey@uni.jena.de)
Marcus Komann (stilgar@minet.uni-jena.de)

Location: University Jena

Short description of the project:

Goal of the project is to use organic computing principles in the architecture of future smart optical detectors. Current architectures for smart optical detectors are not suited to meet hard real-time requirements since they show too centralized structures. Necessary are localised approaches which are scalable independent of the pixel resolution. In our project we want to realise this by utilizing emergent behaviour which is produced by virtual organic units, so-called Marching Pixels. Marching Pixels are a swarm of small directly in hardware realised agents which are crawling within a binary image in order to perform specific tasks, e.g. crawl-ing automatically to centre points of objects by applying only local rules. To find the desired emergent behaviour we pursuit two approaches. The emergence can either be designed by plausibility considerations or it can be evolved by means of an evolutionary algorithm to pro-duce cellular automaton for Marching Pixels. At the end of the project an architecture descrip-tion shall be available for a fast, robust, and scalable smart optical detector chip in which organic computing paradigms are realised.

Application:

Name: Smart cellular real-time CMOS camera for industrial image processing tasks

Description:

In industrial manufacturing and inspection environments it is necessary to detect as fast as possible objects which are lying for example on an assembly line or in a robot cell. The ob-jects have to be identified by their center point and their orientation. Center point and orienta-tion are figures which are essential for a robot which task is it to seize continuously objects. The best way for the robot is to seize the object in the center point in order to minimize the energy costs. E.g. during the inspection in the tool manufacturing industry optical measurements are applied which require the evaluation of up to 1000 images within a second. Another application scenario refers to so-called production assistants. A production assistant is a robot which works together with a human worker. In such a sensitive working environment, consist-ing of co-operation of a robot and a human, one has to consider high security requirements in order to avoid injuries for the human worker. To fulfill these requirements a reply time below

10 ms concerning both hardware and software side is demanded by industry.

In order to meet the mentioned strict real-time requirements more and more high speed microcontrollers and DSPs are currently used. But in principal serial processing schemes will fail to fulfill this ever increasing demands. The drawback of these solutions are increasing size and energy consumption caused by the use of devices which are actually over-dimensioned for the specific tasks of center point and orientation calculation. Increasing size and energy consump-tion are developments which are not tolerable in future embedded systems. Applying organic computing architecture principles in smart optical detectors allows much easier to fulfill the requirements concerning speed, robustness and energy consumption. Speed is achieved by performing mostly local operators in parallel acting agents without usage of time consuming central control units. Robustness is achieved by utilizing emergence on the micro level of pix-els which generates self-organization capabilities observable on the macro level of the whole image. Low energy consumption is achieved since the swarm of agents can be operated with moderate clock cycles on the micro level of pixels. Due to the highly parallel processing of the swarm of agents the real-time requirements on macro level can be met although. That means only the utilization of organic computing principles allows to fulfill the requirements put by industrial application scenarios to a smart detector.

A typical application scenario and the appropriate organic computing hardware which satis-fies this scenario can look like as follows. For example we have given an automated micro-scopic analysis in which the number and the radii of red-blood cell specimen, given as circles in the image, are to determine in a series of input images. Our organic computing smart detec-tor realized as a compact 3D chip stack is positioned in the microscope. The first layer of this chip stack consists of an array of photodiodes and captures the image. The analogue signals are transferred through the first chip layer to the second layer where thresholding and local filter operators are applied to receive a binary image and an edge image of the input image. The tasks in these two first layers comprise the non-organic initial phase. In the next step the binarized input and edge pixels are transferred through the second chip layer to the third layer of the stack. Here our organic Marching Pixels are implemented. Directly in a processor field which is identical with the dimensions of the image, i.e. one input pixel is mapped onto one processor element, the Marching Pixels start to march from the detected edge pixels to the center point. This happens simultaneously in all detected circles. After a fixed time period all Marching Pixels arrived in the center points of the circles and stored there the radius of the circle. Finally an observer processor located in the fourth and last layer of our chip stack fetches the results. The observer processor scans continuously row after row of processing elements in the Marching Pixels processor field. At those positions where a Marching Pixel stopped the number of visited pixels up to the center point is given out along a column line. This value corresponds to the radius of the detected circle. The observer processor can now count the number of circles which are above a certain threshold in order to identify possible ill specimens.

Self-x properties of the application:

In our view organic computing solutions are characterized by self-organization

and emergent capabilities. Both things occur often commonly, but this is not a must, e.g. there are a lot of systems which are self-organizing without using emergence, e.g. a self-organizing operating system which installs automatically a missing driver when a corresponding device is attached.

Emergence: First it is to say that Marching Pixels are more emergent than self-organizing. Nevertheless emergence on the micro level generates often also self-organization capabilities on the macro level. This holds also for our Marching Pixels concept.

Self-reparing: The fail of a certain number pixels is tolerable, since the swarm of Marching Pixels produces a redundancy that can be utilized to increase the robustness of the detection process. This holds for the center point detection in vertical and horizontal reduction lines for symmetrical objects. For details concerning this point we refer to the cited references below.

Self-optimizing: In the same way as ants find the shortest way between food and nest March-ing Pixels will find also the shortest way between two points by using emergent behaviour. If this path is disturbed another shortest path is found automatically. This holds also for the cen-tre point detection.

Other special properties of the Application:

1-stage observer / controller architecture: Many organic computing architec-ture are charac-terized by observer / controller solutions. As described above our organic computing architec-ture for a smart optical detector consists of an processor array for the implementation of the Marching Pixels and an attached observer processor for the post-processing of the results gained by a parallel emergent algorithms. Furthermore the task of the observer processor is the initialisation of the processor elements and to filter out undesired emergence which is caused in our application by noise. In this sense our architecture is characterised by a kind of a one stage observer / controller architecture.

Classification: Industrial Image Processing

Application Reference:

[1] M.Komann, D. Fey: Marching Pixels - Using Organic Computing Principles in Embedded Parallel Hardware, to be published in Proceedings of PARELEC'06 - International Symposium on Parallel Comput-ing in Electrical Engineering, Bialystok, Poland, IEEE CS Press, September 2006

[2]D. Fey, D. Schmidt Marching Pixels: A new organic computing principle for high speed CMOS camera chips Proceedings ACM International Conference on Computing Frontiers 2005, pp. 1-9, Ischia, Italy, May 2005

3 Acknowledgement

Thanks to all the projects in the priority program and its members that wrote a application description for this technical report.