Annual Report 2010
Chair of Computer Science 2 (Programming Systems)

Address: Martensstr. 3, 91058 Erlangen
Phone: +49 9131 85 27621
Fax: +49 9131 85 28809
E-Mail: info@i2.informatik.uni-erlangen.de

Ordinarius:
Prof. Dr. Michael Philippsen
Honorary and Apl. Professors:
Hon.-Prof. Dr.-Ing. Bernd Hindel
Hon.-Prof. Dr.-Ing. Detlef Kips
apl. Prof. Dr.-Ing. Gabriella Kókai, †06.04.2010
Professor Emeritus:
Prof. em. Dr. Hans Jürgen Schneider
Secretary:
Agnes Brütting
Margit Zenk
Scientific Staff:
Dipl.-Inf. Thorsten Blaß
Dipl.-Inf. Daniel Brinkers (from 10.01.2011)
Dipl.-Inf. Georg Dotzler (from 01.02.2010)
Dipl.-Inf. Alexander Dreweke (until 31.01.2011)
Dipl.-Ing. (FH) Thorsten Edelhäußer
Dipl.-Inf. Ralf Ellner
Dipl.-Inf. Philipp Janda
Dipl.-Inf. Stefan Kempf
Dipl.-Inf. Christopher Mutschler (from 15.11.2010)
Dr.-Ing. Norbert Oster
M. Eng. Norbert Tausch, Dipl.-Ing. (FH)
PD Ronald Veldema, Ph.D.
Dipl.-Inf. Tobias Werth
Dipl.-Inf. Marc Wörlein
Guest:
Dipl.-Inf. (FH) Josef Adersberger
Dr.-Ing. Martin Jung
Dr. Karsten Schmidt
External Teaching Staff:
Dr.-Ing. Klaudia Dussa-Zieger

Dipl.-Inf. Stephan Otto

The Chair of Computer Science 2 (programming systems) was founded in 1972 and is headed by Prof. Michael Philippsen (as the successor of Prof. H.-J. Schneider) since April 2002. Closely associated with the programming systems group are the professorship for Didactics of Computer Science and the professorship for Open Source Software.

1 Focus of research

The main research topics in the programming systems group are programming of parallel or distributed systems and programming of embedded or mobile systems. Software (and its development) for such systems should ideally be as complex, portable, maintainable and robust as existing software for single core systems and workstations. It is our long-term goal to allow applications to take full advantage of the available computing and network power. A particular focus lies on programming systems for multi-cores because more and more cheap multi-core high-performance parallel hardware (for example graphics cards or FPA-Hardware) is available. This will have an unpredictable impact on the future of the software landscape. Research results of the group are always evaluated by means of prototypes and demonstrators.

Important Research Areas

- **Exploit the available parallelization potential.** In the future the clock rate of multi-core systems will grow only slowly whereas the number of cores will grow. This makes it necessary to exploit the parallelization potential of already older, existing software to allow it to benefit from the new hardware. As a consequence, in most application areas a change to parallel computing is unavoidable. Therefore, the programming systems group develops tools to support the programmer interactively in reengineering existing sequential applications. It also develops architectural patterns for new software projects that scale automatically to support a growing number of cores.

- **Achieve portability in high-performance applications.** Up to the present, application programmers achieve the best possible performance results only if they handle latency issues and communications between different components of the system manually, optimize their code with hardware specific ”tricks” and split their application into multiple sections to outsource them to other hardware

(for example graphics cards). To change this situation, the programming systems group researches the performance impact of higher programming abstraction layers that would improve programming productivity and software portability. The improvements are caused by generated code that allows the distribution of the program onto multiple heterogeneous system components to permit parallel execution. The higher abstraction layer makes the communication between the components transparent for the developer. To increase the efficiency of this approach it is necessary to give the programmer the possibility to express available domain knowledge in the programming language. For the higher abstraction layer, the details of the hardware architecture are hidden from the developer (for example by library functions or programming language extensions).

- **Adapt the degree of parallelism dynamically**. High-performance applications are often developed for a fixed number of cores. As requested cluster nodes of a batch system are statically assigned for a fixed time period, inefficient reservation gaps are unavoidable. Similar problems appear in multi-threaded applications on multi-core systems. The programming systems group works on the dynamic adaptation of the extent of parallelism by the means of code transformations (under consideration of the resulting data redistribution) and operating system interactions. As control flow based synchronisation measures interfere with the necessary analyzes, the programming systems group researches new programming constructs that can replace the existing ones and allow to specify the synchronisation in a data-centric way.

- **Develop Testing for Parallelism**. In software engineering, testing has always assumed an important role. Code coverage, test data generation, reliability assessment etc. are tools of the trade. Unfortunately, current research insufficiently covers the indeterminism caused by concurrency. To deal with that issue, the programming systems group develops tools that consider (based on the coverage criteria) interleavings of parallel threads in their test data generation. This topic also includes research on operating systems and schedulers. As concurrency considerably increases the search space of the test generation it is necessary to develop infrastructures that allow the test generation and execution on a cluster.

- **Improve of Software Development Processes**. The current development practice of complex, business or security critical software in global distributed teams (commonly found in the software industry) demands compliance with well-defined software development processes. To support the enforcement of this requirement, appropriate development tools are used. The corresponding research area is covered by the Practical Software Engineering research group that is lead

by the honorary professors Dr. Bernd Hindel and Dr. Detlef Kips. Both possess long term experience in industrial software projects as managers of medium sized software companies. The goal of the Practical Software Engineering group is the development of a machine executable notation for modelling of software development processes. For that purpose the research group examines the semi-automatic retrieval of traceability information from the artefacts of different tools and notations as well as the model based development, integration and configurations of software components, used in the design of automotive embedded systems.

2 Research projects

2.1 Compiler-supported parallelization for multi-core architectures

Project manager:
Prof. Dr. Michael Philippsen
Project participants:
Dipl.-Inf. Tobias Werth
Dipl.-Inf. Dominic Schell
Start: 1.3.2007
Contact:
Dipl.-Inf. Tobias Werth
Phone: +49 9131 85-28865
Fax: +49 9131 85-28809
E-Mail: tobias.werth@informatik.uni-erlangen.de

Several issues significantly retard the development of quicker and more efficient computer architectures. Traditional technologies can no longer contribute to offer more hardware speed. Basic problems are the divergent ratio of the latencies of memory access and CPU speeds as well as the heat and waste of energy caused by increasing clock rates.

Homogeneous and heterogeneous multi-core architectures were presented as a possible answer and offer enormous performance to the programmer. The decreasing clock rates help avoid most of the above problems, while the multiplied hardware can still deliver high performance since more arithmetic operations can be executed per time unit with less energy. Potentially, performance can increase even further by specialization of some hardware components. For example, often the latency problem is attacked with a multi-tiered memory hierarchy and lots of caches.

But there is no free lunch. It seems to be quite difficult to make multi-core architectures deliver their theoretically available performance into applications. Only with a lot of expertise in both the application domain and the specifics of the multi-core platform at hand and only with enough time to invest into tuning endeavors, one can make multi-core programs run fast.

From the point of view of a programming systems research group, there are - among others - the following open questions: What kind of support can a modern compiler offer to the programmer that develops applications for multi-core architectures? How much context knowledge is necessary in order to make reasonable decisions for parallelization? Which part of the available performance can be used by the programmer with a reasonable amount of effort without detailed knowledge about the features and quirks of the underlying architecture? Which tools are necessary for debugging and for finding bottlenecks in applications that run on multi-core architectures? How can they be designed?

It is the intention of this new research project to answer these questions for a restricted application domain. We have selected the Lattice-Boltzmann-Method (LBM) that is mostly used in computational fluid dynamics as our problem domain. Caused by its lattice structure and its manageable number of data dependencies between the single lattice points, it is comparatively straightforward how to parallelize it. Hence, our compiler research can focus on the above questions.

The heterogeneous CellBE architecture is selected as target architecture due its good performance on a single chip. It consists of a PowerPC core (PPU) and eight Synergistic Processing Units (SPUs), which can do computations in parallel.

In 2010, the following goals have been on the agenda:

- The programming model Cilk was further developed to allow a robust and efficient execution on the SPUs. We made it much easier to debug the execution while stealing functions from remote SPUs. Beside that, the source to source transformation was rewritten to produce code for both PPU and SPU in a simpler and more generic way.
- Because OpenCL is a good starting point to support other hardware architectures beside the CellBE architecture we have started to parallelize the Lattice Boltzmann Method with OpenCL and tested the parallel code on several architectures. The main research goal was to get to know the details which are abstracted by the available OpenCL implementations of the OpenCL specification and the corresponding hardware without loosing significant performance.
- Furthermore, we have investigated a free library for simulations of the Lattice Boltzmann Method (OpenLB resp. PaLaBoS). This was done by evaluating several approaches for the port and the parallelization that were implemented in

prototypes. Based on that, we are extending the PaLaBoS library so that code can be generated automatically for different models of the Lattice Boltzmann Method in order to execute the simulation on GPUs.

2.2 Graph-based procedural abstraction

Project manager:
Prof. Dr. Michael Philippsen
Project participants:
Dr.-Ing. Alexander Dreweke
Dipl.-Inf. Marc Wörlein
Dipl.-Inf. Tobias Werth
Duration: 1.4.2006–31.12.2010
Contact:
Prof. Dr. Michael Philippsen
Phone: +49 9131 85-27625
Fax: +49 9131 85-28809
E-Mail: philippsen@informatik.uni-erlangen.de

Embedded systems are even nowadays programmed in a machine-oriented fashion. The high level of abstraction and comfort one is used to in the development of desktop applications (object-orientation, garbage collection, exception handling, parallelism, aspect orientation, etc.) seems to be far away for embedded systems. Thus portability, robustness, and maintenance of the software are substantially impaired. This is also an important economical problem because Europe is not yet dominated by the USA in this field. So the long term goal must be to gradually increase the level of abstraction for programming embedded systems, i.e. optimization techniques are needed that achieve small and energy efficient code in spite of the increased abstraction level.

Apart from the obvious question how the well-known standard compiler optimization techniques affect code size, new specific questions arise for embedded systems. While the RAM consumption of an application plays hardly a role on desktop systems, it is crucial for embedded systems. Object-oriented code and especially libraries offer a substantial and mostly unused potential to reduce code size by means of procedural abstraction. Not only code size, but energy efficiency is a target for code optimization on embedded systems. Possibly in cooperation with the operating system, compiler optimizations are crucial. A challenging problem is how to deal with the non-uniform memory hierarchy, that not only consists of registers, caches, and main memory. But in addition, there is another layer, e.g. Flash memory. Another question for embedded systems is wether they can be programmed with the illusion of a uniform memory hierarchy. Is it

possible to extract information about data locality through statical analysis? What kind of optimizations are practicable with this information? Can statical analyzes and runtime mechanisms benefit from each other in this specific area? How can prefetch and post-store commands be generated in the code that help improve latency and energy consumption?

A common method to reduce the size of program code is procedural abstraction (PA): repeated code portions in a program are identified and moved into a single new procedure. Instead of the extracted code portions, calls to the newly created procedure are inserted. This removes redundancy from the program and thus reduces its size. Earlier PA algorithms regarded a program as a sequence of instructions and searched for subsequences that appear frequently. However, if the instruction sequence is altered within a subsequence, they are no longer recognized by sequence matching algorithms. Hence, the result is suboptimal. To solve this problem we transform the instructions of a basic block into a data flow graph (DFG) and use a graph mining tool to search for common fragments in the DFGs of ARM assembly codes that are often used in embedded systems. In cooperation with the project ParSeMiS that is concerned with general optimizations of graph miners, specific properties of the PA domain are exploited. Our research has focused mainly on the analysis optimization of the correct reconstruction of data-flow graphs. The more accurate the data-flow analysis is, the more size reduction can be achieved, compared to the traditional sequential approaches. In addition, various extraction mechanisms have been refined. These mechanisms are used to extract code fragments that have been identified to be frequent by ParSeMiS. The designed extraction mechanisms are as size-efficient as possible and standardize semantic equivalent fragments in a way that they can be extracted together.

2.3 ParSeMiS – the Parallel and Sequential Mining Suite

Project manager:
Prof. Dr. Michael Philippsen
Project participants:
Dipl.-Inf. Marc Wörlein
Dr.-Ing. Alexander Dreweke
Dipl.-Inf. Tobias Werth
Duration: 1.5.2006–31.12.2010
Contact:
Prof. Dr. Michael Philippsen
Phone: +49 9131 85-27625
Fax: +49 9131 85-28809
E-Mail: philippsen@informatik.uni-erlangen.de

The **ParSeMiS** project (**Par**allel and **Se**quential Graph **Mi**ning **S**uite) searches for frequent, interesting substructures in graph databases. This task is becoming increasingly popular because science and commerce need to detect, store, and process complex relations in huge graph structures.

For huge data that cannot be worked on manually, algorithms are needed that detect interesting correlations. Since in general the problem is NP-hard and requires huge amounts of computation time and memory, parallel or specialized algorithms and heuristics are required that can perform the search within time boundaries and memory limits.

Our target is to provide an efficient and flexible tool for searching in arbitrary graph data, to improve the adaption to new application areas, and to simplify and unify the design of new mining algorithms.

In 2010, the distributed stack implementations have also been tested on other algorithms and data structures.

2.4 JavaParty

Project manager:
Prof. Dr. Michael Philippsen
Project participants:
Dipl.-Inf. Marc Wörlein
Start: 1.4.2007
Contact:
Prof. Dr. Michael Philippsen
Phone: +49 9131 85-27625
Fax: +49 9131 85-28809
E-Mail: philippsen@informatik.uni-erlangen.de

[JavaParty]http://svn.ipd.uni-karlsruhe.de/trac/javaparty/wiki/JavaParty allows easy porting of multi-threaded Java programs to distributed environments such as e.g. clusters. Regular Java already supports parallel applications with threads and synchronization mechanisms. While multi-threaded Java programs are limited to a single address space, JavaParty extends the capabilities of Java to distributed computing environments.

The normal way of porting a parallel application to a distributed environment is the use of a communication library. Java's remote method invocation (RMI) renders the implementation of communication protocols unnecessary, but still leads to increased program complexity. The reasons for increased complexity are the limited RMI capabilities and the additional functionality that has to be implemented for creation and access of remote objects.

The JavaParty approach is different. JavaParty classes can be declared as remote. While regular Java classes are limited to a single virtual machine, remote classes and their instances are visible and accessible anywhere in the distributed JavaParty environment. As far as remote classes are concerned, the JavaParty environment can be viewed as a Java virtual machine that is distributed over several computers. Access and creation of remote classes are syntactically indistinguishable from regular Java classes.

We have reached the following goals in 2010:

- Most of the previously self-implemented structures of the run-time system were replaced with more efficient standard Java implementations, because of matters of stability.
- Due to compatibility and security reasons, the communication layer (KaRMI) was reimplemented based on Java's current socket technology.
- A new "near" context was added to remote classes. With this new context that provides host local memory for each instance, locality-aware algorithms can be expressed.

2.5 Jackal – Cluster and Grid computing made easy

Project manager:
Prof. Dr. Michael Philippsen
Project participants:
PD Ronald Veldema, Ph.D.
Start: 1.1.2006

Jackal is a project to create a distributed-shared-memory system for Java. This means that you can write a multi-threaded program (that you could run using normal JVMs on single machines as well) and deploy it on a cluster connected by a network. Jackal also sports a nice check-pointer so it can periodically write the program state to disk for fault tolerance.

To make things more interesting, you can write the program also using OpenMP annotations which allows re-parallelization. Combined with checkpointing this allows a program to be restarted on a different number of machines as that it was started with.

The Jackal-DSM is not only suited for traditional clusters where each node containes only a single CPU and core, but also for newer style clusters where each node in the cluster contains a multi-core CPU. Additionally, Jackal also has extensions and tools for Grid computing.

2.6 Tapir

Project manager:
Prof. Dr. Michael Philippsen
Project participants:
PD Ronald Veldema, Ph.D.
Dr.-Ing. Michael Klemm
Duration: 1.1.2006–31.12.2010
Contact:
PD Ronald Veldema, Ph.D.
Phone: +49 9131 85-27622
Fax: +49 9131 85-28809
E-Mail: veldema@informatik.uni-erlangen.de

Tapir is a new programming language to ease systems programming.

Systems programming includes networking protocols, operating systems, middlewares, DSM systems, etc. Such systems are critical for the functioning of a system as they supply services that are required by user applications. For example, an operating system supplies an operating environment and abstracts from concrete hardware in doing so. A DSM system simulates a single shared memory machine by abstracting from the single machines inside a cluster so that a (user-level application on a) cluster can be programmed without having to program explicit message passing.

Compared to application programming, systems programming has a different set of requirements. The programming 'style' is also very different from the styles used in programming user-level applications. Finally, the performance requirements are usually very strict in systems programs as the complete system's performance greatly relies on the underlying layers of systems programs. Bugs in systems code have also great repercussions on a complete system's reliability. Combined, we can directly conclude the following when using high-level languages for systems programming:

- High-level languages, such as C++, C#, and Java 'hide' implementation details from the programmer. A programmer for example no longer needs to know how exactly a method call is implemented. This knowledge is, however, required when doing systems programming.
- High-level languages supply functionality that is neither required nor wanted. For example, when programming an operating system, automatic language driven exception handling or garbage collection are not wanted.
- Systems programs require no high abstraction levels like common high-level programming languages supply. Likewise, the libraries that a given language offers

can simply not be supplied in a systems context. Usually this is due to the system itself supplying the functionality that the library is to provide.

The basics of the Tapir language have been created. While Tapir has some similarities to Java, C#, and C++, all unneeded and unwanted functionality of the above have been removed. For example, Tapir has no automatic memory management, no exception handling, and no type-casts. Class and object concepts have been kept, but inheritance has been removed. The resulting Tapir programs can be verified by means of model-checking, even while the programming is still being developed. Verification is made easy as code pieces that are implementation details can be marked as such so that they can be safely ignored by the verifier. Tapir code can be executed in parallel for example also on a graphics card without the possibility of the common programming errors associated with parallelism can occur.

Even while the language is still being developed, a prototype DSM protocol has already been implemented in the Tapir language. We have evaluated RDMA-based DSM protocols so that they can be added to the Tapir language. Tapir's semantic checks are implemented by means of model-checking. Model-checking, however, is a very memory intensive analysis. This made us write our own Java Virtual machine, called LVM, which is especially suited for managing large numbers of objects. LVM outperforms standard Java VMs as soon as swapping becomes necessary.

2.7 OpenMP/Java

Project manager:
Prof. Dr. Michael Philippsen
Project participants:
PD Ronald Veldema, Ph.D.
Dr.-Ing. Michael Klemm
Dipl.-Inf. Georg Dotzler
Dipl.-Inf. Thorsten Blaß
Duration: 1.10.2009–1.10.2015

JaMP is an implementation of the well-known OpenMP standard adapted for Java. JaMP allows one to program, for example, a parallel for loop or a barrier without resorting to low-level thread programming. For example:

```
class Test {
...void foo(){
.....//#omp parallel for
.....for (int i=0;i<N;i++) {
```

.....a[i]= b[i]+ c[i]
.....}
...}
}

is valid JaMP code. JaMP currently supports all of OpenMP 2.0 with partial support for 3.0 features, e.g., the collapse clause. JaMP generates pure Java 1.5 code that runs on every JVM. It also translates parallel for loops to CUDA-enabled graphics cards for extra speed gains. If a particular loop is not CUDA-able, it is translated to a threaded version that uses the cores of a typical multicore machine.

In 2010, the JaMP environment was extended to support the use of multiple machines and compute accelerators to solve a single problem. We have developed two new abstraction layers. The lower layer provides abstract compute devices which wrap around the actual CUDA GPUs, OpenCL GPUs, or multicore CPUs, wherever they might be in a cluster. The upper provides partitioned and replicated arrays. A partitioned array automatically partitions itself over the abstract compute devices and takes the individual accelerator speeds into account to achieve an equitable distribution. The JaMP compiler applies code-analysis to decide which type of abstract array to use for a specific Java array in the user's program.

2.8 PATESIA – Parallelization techniques for embedded systems in automation

Project manager:
Prof. Dr. Michael Philippsen
Project participants:
Dipl.-Inf. Stefan Kempf
Dipl.-Inf. Georg Dotzler
Dipl.-Inf. Thorsten Blaß
Start: 1.6.2009
Contact:
Dipl.-Inf. Stefan Kempf
Phone: +49-9131-85-27624
Fax: +49-9131-85-28809
E-Mail: stefan.kempf@informatik.uni-erlangen.de

This project was launched in 2009 to address the refactorization and parallelization of applications used in the field of automation. The programs are executed on specially designed embedded systems. This hardware forms an industry standard and is used worldwide. As multicore-architectures are increasingly used in embedded

systems, existing sequential software must be parallelized for these new architectures in order to gain an improvement of performance.

As these programs are typically used in the industrial domain for the control of processes and factory automation they have a long life cycle. Because of this, the programs are often not being maintained by their original developers any more. Besides that, a lot of effort was spent to guarantee that the programs work reliably. For these reasons, the software is only extended in a very reluctant way.

Therefore, a migration of these legacy applications to new hardware and a parallelization cannot be done manually, as it is too error prone. Thus, we need tools that perform these tasks automatically or aid the developer with the migration and parallelization.

To achieve this, this project works in three areas. The first part examines the migration of legacy applications to new hardware platforms. The challenge here is that the development environments for the new platforms do not support all language constructs that that have been available on the old hardware. This means that the programs need to be refactorized. Second, we examine how the applications can be automatically parallelized after the migration is performed. To guarantee an efficient parallelization, the programs may need to be further restructured. This also required work on the requirements of a runtime environment for parallel programs. We also perform research in a third area that attempts to combine the first two aspects in an expert system that uses a learning approach. The goal is to create a general refactoring tool that can be used for both migration and parallelization.

The development for the migration tool was started in the end of 2010. This subproject treats the problem of the automatic refactorization of code. The main element is a special description language that can be used to describe code patterns that must be refactored. This makes it easy to write rules for refactorings. A pattern is a sequence of statements that should be replaced by some other code sequence. Another possibility is to just inform the user about the occurrence of a pattern if the refactorization cannot be done automatically. The description language, which is still in its development phase, is designed to be easy to use and to be close to natural language. It is our goal to be able to describe complex patterns in a simple and easy to learn way. We started with the design of the language and the integration into a compiler for programming languages used in the field of automation. This compiler is being developed at our department.

It is also used for the second part of our project, which deals with the parallelization of programs. First, we examined applications for automation on their automatic parallelizability. Similar to the migration problem, we found that these programs need to be refactored first, in order to be able to efficiently parallelize them automatically. For this reason, and as a second step, we extended the compiler to search for critical programming constructs that hinder parallelization, and to make the developer aware of them. The patterns we try to find are integrated into the compiler. This part is the origin of the

migration tool described above, where the patterns can be described in a special language. Finally, we continued our work on our runtime environment for parallel programs, which we started in 2009. This library dynamically distributes parallel tasks that are created at runtime over the available cores. This mechanism was made more efficient by an internal reorganization, in order to evenly balance the work over all cores. Besides that, an implementation of transactional memory and a model for interrupt handling was added to the library as well.

We started to develop a learning expert system in 2010. This system is able to assist developers in optimizing and restructuring their programs. It is our goal to not have to specify the patterns and their replacements by hand. Instead, the system should compare original and modified codes and learn by itself, which code transformations were applied. As a first step, we developed parsers for several programming languages, like C and Java. They read program source code and generate a language independent abstract syntax tree (AST). After that, we created a prototype that finds improvable or refactorizable code locations in those syntax trees. If a pattern in the AST is found, our system generates the refactored code and presents them to the user as suggestions that are generated by using appropriate patters from a database. The patterns are generated by comparing original and modified source codes and then fed into the database.

2.9 Model Driven Component Composition

Project manager:
Prof. Dr. Michael Philippsen
Project participants:
Dipl.-Inf. Philipp Janda
Duration: 15.6.2007–14.6.2011
Sponsored by: AUDI AG

This project is part of the INI.FAU collaboration between AUDI AG and the University of Erlangen-Nuremberg. It examines model-driven ways to integrate vehicle functions on electronic control units (ECUs). Moreover, the project develops supporting methods and tools for this task. The insights gained in the course of this project will be practically validated by integrating a damper control system into an AUTOSAR ECU.

In the automotive industry it is common practice to develop in-car-software on a high level of abstraction and in a model-based way. To eliminate uncertainties concerning resource consumption and runtime it is necessary to test the developed software on the target hardware as early as possible. But due to cost and safety requirements the integration of the software into an ECU is very time-consuming and demands special expertise going beyond that of the function developer. AUTOSAR (AUTomotive Open

Systen ARchitecture) is on the way to become a standard for the basic software on ECUs. But due to the novelty of this standard there are neither processes nor tools to support the integration of the developed in-car-software into an ECU.

In 2008, we have examined the modeling expressiveness of AUTOSAR with respect to both its applicability and possible conflicts with existing standards and technologies that are currently in use at Audi. Furthermore, the automatic generation of an AUTOSAR software architecture from a single damper control component has been implemented.

Since 2009, a model-driven approach that supports the integration of software into an ECU is being implemented and integrated into the tool set used at Audi. In particular we are looking at the automatic configuration of the bus communication by means of a bus database and the automatic task scheduling among the application processes. The model-driven development, which in this case is based on the Eclipse Modeling Framework, will enable easier tayloring of the emerging prototype to changing requirements.

In 2010, we exploited the information that is available in an AUTOSAR project to automatically configure local and remote communication between software components. We have also developed a genetic algorithm that uses dependency information to automatically generate task schedulings that minimize communcation latencies between cooperating software components. The existing prototype has been extended with the above-mentioned methods.

2.10 Integrated Tool Chain for Meta-model-based Process Modelling and Execution

Project manager:
Hon.-Prof. Dr.-Ing. Detlef Kips
Project participants:
Dipl.-Inf. Ralf Ellner
Prof. Dr. Michael Philippsen
Dr.-Ing. Martin Jung
Dipl.-Inf. Johannes Drexler
Al-Hilank, Samir
Duration: 1.10.2008–30.9.2011
Sponsored by: Bundesministerium für Wirtschaft und Technologie

As demands on the development of complex software systems are continuously increasing, compliance with well-defined software development processes becomes even more important. Especially large and globally distributed software development projects tend to require long-running and dynamically changeable processes spanning

multiple organizations. In order to describe and support such processes, there is a strong need for suitable process modeling languages and for powerful support by tools.

The results of a preceding cooperation project show that today's tools markets lack integrated tool chains which actually support the fine-grained and precise modeling of software development processes as well as their computer-aided execution, controlling and monitoring. An ongoing cooperation project intends to bridge this gap. This cooperation project is carried out together with develop group as an industrial partner and is funded by BMWi. It started in October 2008 and has been scheduled for three researchers over a period of three years.

The objective of this cooperation project is to prototype an integrated tool chain by using a rigorous, meta-model based approach which supports modeling, enactment and execution of industrial software development processes. Bearing the applicability of such a tool in mind, this approach is mainly intended to provide a wide adaptability of process models to different industrial development scenarios, to define a user-friendly concept of process description and to establish an extensive computer-aided process execution support, contributing to the efficiency of development activities. These benefits will be achieved by a high grade of formalism, by an integrated, generic concept of process modeling and process enactment and by using commonly accepted industrial standards (UML, SPEM).

The integrated tool chain developed in this project is based on an extension of the SPEM standard (eSPEM – enactable SPEM). eSPEM adds a behaviour modelling concept by reusing UML activity and state machine diagrams. In addition, eSPEM provides behaviour modelling concepts that are specific to software development processes, for example, dynamic task creation and scheduling. An overview of these SPEM extensions has been presented at the "6th European Conference of Modelling Foundations and Applications".

Within 2010, the existing prototype of an eSPEM-based process modeling environment has been significantly extended. Using this prototype, eSPEM has been validated by modeling development processes which are widely used in industrial development teams. As a result, eSPEM has been further enhanced and improved with respect to its usability for real life projects and processes. A prototype of a process execution environment has been implemented, which is able to instantiate, simulate, and execute eSPEM-based process models in distributed environments. The process execution environment has been validated by executing exemplary software process models.

2.11 Softwareleitstand – Prototypical implementation of a new tool for quality assurance during software development

Project manager:
Prof. Dr. Michael Philippsen
Project participants:
Dipl.-Inf. (FH) Josef Adersberger
M. Eng. Norbert Tausch, Dipl.-Ing. (FH)
Start: 1.11.2009
Sponsored by: Bundesministerium für Wirtschaft und Technologie
Contact:
Prof. Dr. Michael Philippsen
Phone: +49 9131 85-27625
Fax: +49 9131 85-28809
E-Mail: philippsen@informatik.uni-erlangen.de

Modern software systems are getting increasingly complex with respect to functional, technical and organizational aspects. Thus, the number of requirements per system and the degree of their interconnection constantly rise. Furthermore the technical parameters, e.g., for distribution and reliability are getting more complex and software is developed by teams that are not only spread around the globe and also suffer from increasing time pressure. Due to this, the functional, technical, and organizational control of software development projects is getting more difficult.

The ”Softwareleitstand” is a tool for managing project roles like the project leader, the software architect, the requirements engineer, or the head of development. Its purpose is to make all aspects of the development process transparent and thus to allow for better project control. To achieve transparence, the tool distills and gathers properties from all artifacts and correlations between them. It presents/visualizes this information in a way suitable for the individual needs of the users.

The Softwareleitstand unifies the access to relations between artifacts (traceability) and to their properties (metrics) within software development projects. Thus, their efficiency can be significantly increased. The artifacts, their relations, and related metrics are gathered and integrated in a central data store. This data can be analyzed and visualized, metrics can be computed, and rules can be checked.

The project ”Softwareleitstand” is performed in cooperation between the programming systems group at the University of Erlangen-Nuremberg and the QAware GmbH, Munich. It is funded by the AIF ZIM program of the German Federal Ministry of Economics and Technology. The project period is from november 2009 to may 2012.

The Softwareleitstand is devided into two subsystems: The integration pipeline which gathers traceability data and metrics from a variety of software engineering tools, and the analysis core, which allows to analyze the integrated data in a holistic way. Each subsystem is developed in a separate subproject.

The integration pipeline is implemented by the project partner QAware GmbH. The first step was to define TraceML, a modeling language for traceability information in conjunction with metrics. The language contains a meta-model and a model library. TraceML allows to define customized traceability models in an efficient way. The integration pipeline is realized using TraceML as lingua franca in all processing steps: From the extraction of traceability information to its transformation and integrated representation. Eclipse Modeling Framework is used to define the TraceML models on each meta-model level. Furthermore, the Modeling Workflow Engine is used for model transformations and Eclipse CDO is used as model repository. A set of wide-spread tools for software engineering are connected to the integration pipeline including Subversion, Eclipse, Jira, Enterprise Architect and Maven.

The analysis core is implemented by the research partner. Within the report period, a concept for the analysis core was elaborated and its implementation was started. The analysis core provides a mechanism to query the traceability information which is provided as an object graph by the integration pipeline. These queries are defined in a domain-specific query language (DSL) for traceability information leveraging concepts of the graph theory. A study was performed to examine the usability of already existing languages in the domain of graph processing and traceability information processing. This lead to the decision to implement an own domain specific language based on the functional and object-oriented multi-paradigm language Scala. The resulting DSL has reached its first stage and is capable of creating so-called Reflexion Models which are representing a mapping between component and code models in form of convergence and divergence relations. Furthermore, the aggregation of metrics from the code model to the component model is supported.

2.12 Wireless Localization

Project manager:
Prof. Dr. Michael Philippsen
Project participants:
Dipl.-Ing. (FH) Thorsten Edelhäußer
Dipl.-Inf. Christopher Mutschler
Duration: 1.5.2008–14.11.2013
Sponsored by: Fraunhofer Institut für Integrierte Schaltungen

Localization Systems (also known as Realtime Location Systems, or RTLS) become more and more popular in industry sectors such as logistics, automation and many more. These systems provide valueable information about whereabouts from objects at runtime. Therefore, processes can be traced, analyzed and optimized. Besides the research activities at the core of localization systems (like resilience and interference-free location technologies or methods for highly accurate positioning), there emerge algorithms and techniques to identify meaningful information for further processing steps. In this context, the aim of the wireless localization project is the research on automatic configuration methods for RTLSs as well as the generation of dynamic moving models and techniques for event processing on position streams at runtime.

In 2009, we continued the development of our algorithms to estimate the receiving antenna's position (pose) of location systems. The algorithms estimate measuring points which allow a fast and accurate measurement pose. We applied an robot to execute the measurement automatically. The developed algorithm considers obstacles and the receiving characteristics of the location system and can sort out errors contained in the measurement data (e.g. multipath measurements).

In 2010, models have been developed that can be used as dynamic moving models. Learning methods are applied to adapt the models at run-time. A formal language called TBL (Trajectory Behavior Language) was developed for describing trajectories. Additional algorithms can shrink the size of that description and hence compress the data size required to store trajectories.

We are evaluating methods for learning the moving models online. These are evaluated in a study with respect to motion prediction. Moreover, it is being investigated whether events can be predicted by analyzing and learning event streams from the localisation system at runtime.

2.13 Graphs and Graph Transformations

Project manager:
Prof. em. Dr. Hans Jürgen Schneider
Duration: 1.10.2004–30.9.2012
Contact:
Prof. em. Dr. Hans Jürgen Schneider
Phone: +49 9131 85-27620
Fax: +49 9131 85-28809
E-Mail: schneider@informatik.uni-erlangen.de

Graphs are often used as an intuitive aid for the clarification of complex matters.

Outside the field of computer science this is for example true in biology or chemistry, where molecules are modeled in a graphical way. In computer science, data or control flow charts are often used as well as entity relationship charts or Petri-nets to visualize software or hardware architectures. Graph grammars and graph transformations combine ideas from the fields of graph theory, algebra, logic, and category theory, to formally describe changes in graphs.

The underlying theory is an attractive tool for the description of extremely different structures in a uniform way, e.g., the different models for asynchronous processes: Petri-Nets are based on standard labeled graphs, state charts use hierarchical graphs, parallel logic programming can be interpreted in a graph-theoretical way using so-called jungles, and the actor systems can be visualized as graphs, whose labeling alphabet is a set of term graphs.

In 2010, we have again concentrated our attention on theoretical concepts as well as on an implementation aspect:

- The focus of our research on graph transformation is the so-called double-pushout approach in which a production is given by two morphisms starting at a common interface graph. The ongoing work on the planned textbook has lead to analyze some alternatives and to study the relationships between them and the double-pushout approach. We have first considered hyperedge replacement and inward productions. Hyperedge replacement is well-suited to formally describe computations as well as graphical representations. In the literature, it is usually not treated in the context of the double-pushout approach since the mainstream of DPO research excludes non-injective mappings. Our previous results have avoided this restriction and, therefore, hyperedge replacement also fits well into the double-pushout approach. As a side effect, we find a class of context-free graph grammars that includes some languages that are not contextfree when treated as string grammars. The second point is based on a proposal by Banach. He has reversed the directions of the morphisms in the production such that they go inwards and embed the left-hand side and the right-hand side into a larger context. This approach allows establishing some interesting relations to operational (non-categorical) approaches as well as to the adhesive-grammar approach. The details will be soon included into the draft of our textbook: http://www2.informatik.uni-erlangen.de/Personen/schneide/gtbook/chapter6.pdf

- The categorical approach to graph transformations is highly generic: All the proofs and constructions are valid for various types of graphs. Only the basic operations must be described for each application in detail, the categorical properties on top of these are then defined automatically. Since modern programming languages support generic concepts, it is a promising idea to implement the categorical approach to graph transformation in languages such as Java or Haskell.

In 2008, we have implemented this approach in the functional language Haskell taking advantage of polymorphism. Java uses classes of objects, but does not really support multiple inheritance. In our 2009 version, we have defined interfaces Cat, CatWithColimits, etc. which must be implemented by classes such as CatFinSet, CatFinGraph. This solution, however, does not allow us to provide CatWithColimits with a method constructing pushouts. Therefore, we defined a separate factory class PushoutCreator describing this categorical construction independent of a special category, but it must be imported by the classes implementing special categories. Now, we have implemented an alternative based on generic Java data types. The trick is to define an abstract class CocoCat of cocomplete categories and to provide this class with the canonical constructions of colimits, e.g., pushouts, making them available in the concrete subclasses. The necessity to decorate all the classes and all the methods with the generic parameters is the main disadvantage of this solution. A summary of this implementation is published in Engels et al. (Eds.): Graph Transformation and Model-Driven Engineering (Lect. Notes Comp. Sc. vol. 5765, 2010), pp. 33-58.

2.14 International Collegiate Programming Contest at the FAU

Project manager:
Prof. Dr. Michael Philippsen
Project participants:
Dipl.-Inf. Tobias Werth
Dipl.-Inf. Marc Wörlein
Dr.-Ing. Alexander Dreweke
Dipl.-Inf. Daniel Brinkers
Start: 1.11.2002
Contact:
Dipl.-Inf. Tobias Werth
Phone: +49 9131 85-28865
Fax: +49 9131 85-28809
E-Mail: tobias.werth@informatik.uni-erlangen.de

The Association for Computing Machinery (ACM) has been hosting the International Collegiate Programming Contest (ICPC) for many years. Teams of three students try to solve nine to ten programming problems within five hours. What makes this task even harder, is that there is only one computer available per team. The problems demand for solid knowledge of algorithms from all areas of computer science and mathematics, e.g. graphs, combinatorics, strings, algebra and geometry.

The ICPC consists of three rounds. First, each participating university hosts a local con-

test to find the three teams that are afterwards competing in one of the various regional contests. Erlangen lies in the catchment area of the Northwestern European Regional Contest (NWERC) where also teams from e.g. Great Britain, Benelux and Scandinavia. The winners of all regionals in the world (and some second place holders) advance to the world finals in spring of the following year.

In 2010 two local contests took place in Erlangen. During the winter semester a team contest was conducted with teams consisting of at most three students. The main goal of this contest was to interest new students in the contests. There has been a record attendance of 24 FAU teams. Also 13 teams from TU Munich and 2 teams from the University of Konstanz competed with our teams online.

As in the previous years, in the summer term the seminar "Hello World - Programming for Advancers" served to prepare students from different disciplines in algorithms and contest problems. In the contest of the summer term, which was organized as national contest for the first time, the representatives of the FAU Erlangen-Nuremberg for the NWERC 2010 in Bremen were chosen. 15 teams with students of computer science, computational engineering, mathematics as well as informations and communication technology took part in the contest. Ten students were selected for the NWERC forming three teams and one reserve. At the NWERC in Bremen, our best team won clearly against the competitors from Northwestern Europe. The second team also did a great job resulting in the fourth place. With a rank in the middle (33rd of 67 teams) for the third team, the NWERC was a great success for the FAU. It seems that again, the trainings-camp that resulted in the ticket to the World Finals in Sharm-el-Sheikh/Egypt, served as a perfect preparation.

2.15 Embedded Systems Institute

Project manager:
Prof. Dr. Michael Philippsen
Project participants:
Dipl.-Inf. Philipp Janda
Dipl.-Inf. Stefan Kempf
Dipl.-Inf. Georg Dotzler
Dipl.-Inf. Thorsten Blaß
Start: 1.9.2007

In September 2007 the ESI – Embedded System Institute – was founded as an interdisciplinary center at the Friedrich-Alexander-University (FAU) with the goal to coordinate and organize research, teaching, and further education in the field of embedded systems.

ESI brings together existing skills within the university and interests, activities, and goals of large and medium sized companies in the field of embedded systems.

Companies obtain access to latest research results and the opportunity to develop common projects, to establish ties, and to find co-operation partners. The ESI concentrates the skills of the chairs of computer sciences and makes them usable for co-operation projects. Hence, the latest research results can be transfered into products in a speedy way. Finally, the ESI may serve as a platform for recruiting excellent students and highly qualified young academics at an early stage.

The chair of computer science department 2 (Prof. Philippsen) is one of the active founders of the ESI and carries out research projects within the ESI.

More information can be found at http://www.esi.uni-erlangen.de and http://www.esi-anwendungszentrum.de

3 Publications

– Dotzler, Georg ; Veldema, Ronald ; Klemm, Michael: JCudaMP: OpenMP/Java on CUDA . In: Pankratius, Victor ; Philippsen, Michael (Ed.) : Proceedings of the Third International Workshop on Multicore Software Engineering (IWMSE10) (International Workshop on Multicore Software Engineering Cape Town, South Africa 01.05.2010). Los Alamitos, CA, USA : ACM Press, 2010, pp 10-17. - ISBN 978-1-60558-964-0

– Dreweke, Alexander: Graphbasierte Prozedurale Abstraktion . Erlangen, Friedrich-Alexander-Universität, Ph.D. thesis, 2010. - 201 pages.

– Edelhäußer, Thorsten ; Frühauf, Hans-Holm: Concept for encoding data defining coded orientations representing a reorientation of an object . Schutzrecht 10196868.3 DE patent application (23.12.2010)

– Edelhäußer, Thorsten ; Frühauf, Hans-Holm ; Kókai, Gabriella ; Philippsen, Michael: Concept for encoding data defining coded positions representing a trajectory of an object . Schutzrecht 10196851.9 DE patent application (23.12.2010)

– Edelhäußer, Thorsten ; Kókai, Gabriella ; Frühauf, Hans-Holm: Design for determining an estimated value for a position of a reception element . Schutzrecht WO/2010/128121 DE examined granted patent (11.11.2010)

– Edelhäußer, Thorsten ; Janak, Mateusz ; Kókai, Gabriella: Environment-Based Measurement Planning for Autonomous RTLS Configuration . In: IEEE (Ed.) : Proceedings of the Conference on Adaptive Hardware and Systems (AHS 2010)

(Conference on Adaptive Hardware and Systems (AHS 2010) Anaheim California, USA). 2010, pp 289-296.

– Ellner, Ralf ; Al-Hilank, Samir ; Drexler, Johannes ; Jung, Martin ; Kips, Detlef ; Philippsen, Michael: eSPEM - A SPEM Extension for Enactable Behavior Modeling . In: Kühne, Thomas ; Selic, Bran ; Gervais, Marie-Pierre ; Terrier, Francois (Ed.) : 6th European Conference of Modelling Foundations and Applications (ECMFA 2010, 6th European Conference of Modelling Foundations and Applications Paris, France June 15-18, 2010). Vol. 6138. Berlin : Springer, 2010, pp 116-131. (Lecture Notes in Computer Science Vol. 6138) - ISBN 978-3-642-13594-1

– Haumacher, Bernhard ; Philippsen, Michael ; Tichy, Walter F.: Irregular data-parallelism in a parallel object-oriented language by means of Collective Replication . Erlangen : Universität Erlangen, Dept. of Computer Science 2. 2010 (CS-2010-04). - Internal report. 14 pages (Department Informatik - Technical Reports Vol. CS-2010-04) ISSN 2191-5008

– Schneider, Hans Jürgen ; Minas, Mark: Graph transformation by computational category theory . In: Engels, Gregor ; Lewerentz, Claus ; Schäfer, Wilhelm ; Schürr, Andy ; Westfechtel, Bernhard (Arr.): Graph Transformations and Model-Driven Engineering - Essays Dedicated to Manfred Nagl on the Occasion of his 65th Birthday. Berlin/Heidelberg : Springer, 2010, (Lecture Notes Computer Science Vol. 5765), pp 33-58. - ISBN 978-3-642-17321-9

– Veldema, Ronald: Improved DSM Efficiency, Flexibility, and Correctness . Erlangen, FAU, Habil. thesis, 2010 (Technical reports / Department Informatik Vol. CS-2010,3) . - 144 pages. ISSN 2191-5008

– Veldema, Ronald ; Philippsen, Michael: Safe and Familiar Multi-core Programming by means of a Hybrid Functional and Imperative Language . In: Gao, Guang R. ; Pollock, Lori L. ; Cavazos, John ; Li, Xiaoming (Ed.) : Languages and Compilers for Parallel Computing (22nd International Workshop of Languages and Compilers for Parallel Computing (LCPC 2009) Delaware October 8-10, 2009). Vol. 5898. Berlin : Springer, 2010, pp 157-171. - ISBN 978-3-642-13373-2

– Werth, Tobias: DOMjudge - An Automated Judging System for Programming Contests .Talk: Collaborative Learning Institute Symposium, Collaborative Learning Institute, Harbin, China, 04.02.2010

– Zvada, Szilvia: Attribute Grammar Based Genetic Programming . Erlangen, FAU, Ph.D. thesis, 2010. - 213 pages.

4 Exam theses (german only)

– Studienarbeit: Automatisches und integriertes Testen von elektronischen Bauelementen. Bearbeiter: Eberlein Stefan (beendet am 20.01.2010); Betreuer: PD Ronald Veldema, Ph.D.; Dipl.-Ing. Sabine Walther

– Diplomarbeit: Definition von Entwicklungsprozessen für Kollaborationsplattformen am Beispiel von IBM Rational Jazz und Team Concert. Bearbeiter: Gabriel Dexheimer (beendet am 03.02.2010); Betreuer: Hon.-Prof. Dr.-Ing. Bernd Hindel; Prof. Dr. Michael Philippsen

– Studienarbeit: Interaktive Prozesssteuerung für ein verteiltes Zeitplanungssystem. Bearbeiter: Kerim Merdenoglu (beendet am 16.02.2010); Betreuer: PD Dr.-Ing. habil. Peter Wilke

– Diplomarbeit: Carolo-Cup: Bewegungsplanung für das Fahrzeug. Bearbeiter: Philipp Sommer (beendet am 17.2.2010); Betreuer: apl. Prof. Dr.-Ing. Gabriella Kókai; Prof. Dr. Michael Philippsen

– Studienarbeit: Case-Study: Untersuchung des industriellen Einsatzes von .gEAr im web-basierten Client-Server-Umfeld. Bearbeiter: Renato Gabriel Giurea (beendet am 01.4.2010); Betreuer: Dr.-Ing. Norbert Oster; Prof. Dr. Michael Philippsen

– Diplomarbeit: Mehrstufige Parallelisierung von Graph-Mining-Algorithmen. Bearbeiter: Bernd Schöbel (beendet am 03.05.2010); Betreuer: Dipl.-Inf. Marc Wörlein; Prof. Dr. Michael Philippsen

– Diplomarbeit: Online Data-Mining of trajectories in Real Time Location Systems. Bearbeiter: Christopher Mutschler (beendet am 12.05.2010); Betreuer: apl. Prof. Dr.-Ing. Gabriella Kókai; Prof. Dr. Michael Philippsen; Dipl.-Ing. (FH) Thorsten Edelhäußer

– Studienarbeit: Konzeptionierung und Pilotierung eines automatisierten Deploy- und Test-Systems. Bearbeiter: Fabian Zach (beendet am 01.07.2010); Betreuer: Dr.-Ing. Norbert Oster; Prof. Dr. Michael Philippsen

– Diplomarbeit: Reengineering des modellbasierten Testdatengenerators UnITeD hinsichtlich der verwendeten Heuristiken zwecks Optimierung der Performance und Ergebnisqualität. Bearbeiter: Kai Guo (beendet am 26.07.2010); Betreuer: Dr.-Ing. Norbert Oster; Prof. Dr. Michael Philippsen

– Diplomarbeit: Multi-GPU Cluster use for Java/OpenMP. Bearbeiter: Thorsten Blaß (beendet am 02.08.2010); Betreuer: PD Ronald Veldema, Ph.D.; Prof. Dr. Michael Philippsen

– Diplomarbeit: Analytical and Experimental Evaluation of SMP Superscalar. Bearbeiter: Jan Ciesko (beendet am 2.11.2010); Betreuer: PD Ronald Veldema, Ph.D.; Prof. Dr. Michael Philippsen