08281 Abstracts Collection
Software Engineering for Tailor-made Data Management
— Dagstuhl Seminar —

Sven Apel[1], Don Batory[2], Goetz Graefe[3], Gunter Saake[4] and Olaf Spinczyk[5]

[1] Universität Passau, D
apel@uni-passau.de
[2] Univ. of Texas at Austin, USA
[3] Hewlett Packard Labs- Palo Alto, USA
[4] Universität Magdeburg, D
saake@iti.cs.uni-magdeburg.de
[5] Technische Universität Dortmund, D
Olaf.Spinczyk@tu-dortmund.de

Abstract. From July 6th to July 11th, 2008, the Dagstuhl Seminar 08281 "Software Engineering for Tailor-made Data Management" was held in the International Conference and Research Center (IBFI), Schloss Dagstuhl. During the seminar, several participants presented their current research, and ongoing work and open problems were discussed. Abstracts of the presentations given during the seminar as well as abstracts of seminar results and ideas are put together in this paper. The first section describes the seminar topics and goals in general. Links to extended abstracts or full papers are provided, if available.

Keywords. Software Engineering, Data Management, Software Product Lines, Embedded Systems

08281 Executive Summary – Software Engineering for Tailor-made Data Management

Tailor-made data management software (DMS) is not only important in the field of embedded systems. DMS that incorporates only features that are required bear the potential to strip down the code base and to improve reliability and maintainability. In the past 20 years several new technologies have emerged that aim at lean, efficient, and well-structured software, which should also be applicable to DMS. In the Dagstuhl Seminar "Software Engineering for Tailor-made Data Management", July 6h to July 11th, 2008, 29 researchers from 7 countries discussed the development, application, and assessment of these new technologies in the context of DMS.

Keywords: Software Engineering, Data Management, Tailoring, Embedded Systems

Joint work of: Sven Apel; Don Batory; Goetz Graefe; Gunter Saake; Olaf Spinczyk

Extended Abstract: http://drops.dagstuhl.de/opus/volltexte/2008/1578

KALA: Helping the application programmer make sense of advanced transactions

Johan Fabry (DCC - Universidad de Chile, CL)

In the eighties and nineties, a significant amount of research had already been performed with regard to tailor-made data management, more specifically in the domain of advanced transaction models. Advanced transaction models go beyond the classical ACID transaction concepts to provide a better mapping of application behavior to concurrent data access, allowing, amongst others, for higher performance. Ten years later, we see that none of this research has had any impact. We are hard-pressed to find a correct implementation of the best-known advanced transaction model: nested transactions, in the real world.

So what happened? Why is this research, which is becoming ever more relevant, not being used? Is it because it is too difficult to implement the servers that provide support for these advanced models? We think not, as multiple academic implementations of such servers exist. Our position is that these models have not caught on because they are simply too difficult to use for the average application programmer. In a world where understanding classical transactions is already considered an 'advanced topic', where should we classify advanced transaction models?

Our research statement is therfore the following: The bottleneck for the use of tailor-made data management is the complexity that this exposes to the application programmer. In order for this topic to be succesful we need to simplify his/her life when using tailor-made data management.

We present a case-study in the form of KALA: an aspect language that is specifically designed to express advanced transaction models, providing abstractions at a less painful level. Furthermore, we show how KALA allows for model-specific languages to be easily defined and implemented. This provides for a maximal level of abstraction, enabling an 'Using advanced transaction models for dummies' approach to tailor-made data management.

Tailoring Data-Intensive Systems via Source Code Generation

Bernd Fischer (Univ. of Southampton, GB)

Data-intensive systems consist of some application code on top of a DBMS. Tailoring such systems involves tailoring the application code at least as much as the underlying DBMS.

I argue that this application code tailoring can be done by source code generation from high-level models, and present some examples of generator application. I also show an approach to ensure the correctness of the generated code that does not rely on the correctness of the generator itself.

Keywords: Code generation

MXQuery: Scaling XQuery from Sensor Motes to the Cloud

Peter Fischer (ETH Zürich, CH)

XQuery provides a turing-complete expression, declarative and optimizable language to work on many types of data, not only XML. Extensions are available towards the areas of fulltext procesing, updates, stream processing and general programming. One important potential use of XQuery is to enable cross-platform, cross-layer optimizations since the same language is used all over the stack/system.

To enable such possibilities, we demonstrate how an XQuery implementation can be scaled down towards devices using the Java Micro Edition and a limited amount of memory.

Keywords: XQuery

FeatureMapper: Mapping Features to Models

Florian Heidenreich (TU Dresden, DE)

Ongoing research in Software Product Line Engineering (SPLE) emphasises the derivation of a concrete product based on a given variant configuration as one of the most promising areas of the field. To allow for (automatic) derivation of products in SPLE, models that describe features and their variability in Software Product Lines (SPLs)—for example feature models—need to be connected with artefacts that are realising the features.

It is crucial not only to support the developer in the complex task of defining such connections, but also to provide means to reason and analyse them—for example visualisations. In this talk we present FeatureMapper, a tool that allows to define mappings between variability models (e.g. feature models) and solution models (e.g. UML models), provides different visualisations of those mappings and allows for mapping-based transformation.

Keywords: Software Product Lines, Feature Modelling, Feature Visualisation, Feature Mapping, Model Transformation

Joint work of: Heidenreich, Florian; Kopcsek, Jan; Wende, Christian

Tailor-made Support of Transactions in Native XML Databases

Theo Härder (TU Kaiserslautern, DE)

Apparently, everything that can be said about concurrency control and recovery is already said. None the less, the XML model poses new problems for the optimization of transaction processing. In this talk, we report on our view concerning XML transaction optimization. We explore aspects of fine-grained transaction isolation using tailor-made lock protocols. Furthermore, we outline XML storage techniques where storage representation and logging can be minimized in specific application scenarios

Keywords: XML database management, concurrency control, logging and recovery, elementless XML storage

Joint work of: Härder, Theo; Bächle, Sebastian; Mathis, Christian

Tailoring Middleware

Hans-Arno Jacobsen (University of Toronto, CA)

Middleware becomes increasingly important in building distributed applications. Today, conventional middleware systems are designed, implemented, and packaged prior to their applications. We argue that with this middleware construction paradigm it is often difficult to meet the challenges imposed by application specific customization requirements.

We propose to reverse this paradigm by automatically synthesizing middleware structures as the result of reasoning about the distribution needs of the user application of middleware.

We term this type of post-postulated middleware Just-in-time middleware (JiM). In this paper, we present our initial design and present an evaluation of the JiM paradigm through Abacus, a CORBA middleware implementation based on the aspect oriented refactoring of an industrial strength object request broker. In addition, we present Arachne, the Abacus synthesizer, which integrates source analysis, feature inference, and implementation synthesis. Our evaluations show that, through automatic synthesis alone, Abacus is able to support diversified application domains with very flexible architectural compositions and versatile resource requirements as compared to conventional pre-postulated approaches.

Keywords: Aspect Oriented Middleware, Middleware Architecture

Full Paper:
http://www.eecg.toronto.edu/~jacobsen

Workload Management in Business Intelligence Database Systems

Stefan Krompaß (TU München, DE)

Business Intelligence workloads that run against very large data warehouses contain queries whose execution times range, sometimes unpredictably, from seconds to hours. The presence of even a handful of long-running queries can significantly slow down a workload consisting of thousands of queries, creating havoc for high priority queries that require a quick response. We devise an experimental framework that (1) can recognize long-running queries and categorize them in terms of their impact on performance, (2) lets us experiment with different workload management policies devised in academia and industry, including policies not yet implemented in commercial database systems, and (3) lets us replay actual workloads and also insert problem queries into them.

Keywords: Workload management, experimental framework, execution engine model

Decomposing Berkeley DB – Implementation, Granularity, and Interactions

Christian Kästner (Universität Magdeburg, DE)

Before designing and implementing a database software product line it is necessary to understand the role of features in database system. Therefore, we conducted a case study in refactoring an existing commercial-quality embedded database system – Berkeley DB – into features. During the decomposition we analyzed how features were implemented in the legacy code, and which language constructs are necessary to decompose them.

We found that a features is implemented by a combination of multiple extensions to the program of different granularity (from adding a class to inserting a statement in the middle of a method) which cut accross the entire implementation. There is no clean separation of concerns due to performance and memory concerns.

We implemented the decomposition both with AspectJ as a representative of compositional approachs (generating a variant by composing code) and CIDE as a represenative of annotative approaches (generating a variant by removing annotated code). However, both solutions have advantages and drawbacks – from usability and readability to feature interaction problem. To address these problems we propose additional tool support.

Keywords: Berkeley DB, FOP, CIDE, Preprocessors, Feature Interactions

Joint work of: Kästner, Christian; Apel, Sven; Batory, Don; Saake, Gunter

Data management for robotics

Jörg Liebig (Universität Magdeburg, DE)

The upcoming roboter technology faces the challenges of increased functionality and the interaction with humans.

Along with these challenges the amount of data that has to be gathered and stored rises.

Our research addresses tailor-made programming for a varity of hardware platforms that form a product line. Since it seems to be easier to add functionality rather than taking it out of a system, we define a lower bound of embedded systems that need data management functionalities and scale to larger systems from there.

Keywords: Data management, robotics, embedded systems

The Storage System of a Runtime Adaptable DBMS

Klaus Meyer-Wegener (Universität Erlangen, DE)

Experience during the last decades has shown that database management systems (DBMS) often have lifetimes over 20 or 30 years. The surrounding environment can change dramatically in this period of time while the system is in use. Furthermore often a lot of different applications depend on a single DBMS. Shutting down the DBMS for updates or bug fixes affects all dependent applications and this is not volitional or even feasible in some environments.

In this talk we present the CoBRA DB (component based runtime adaptable database) project that is developed on basis of a service-oriented component model. The flexible design permits the development of tailor-made DBMSs by selecting the appropriate services for a specific environment respectively application. Due to this fact, unnecessary components are not composed into the DBMS, resulting in a small footprint. Furthermore we introduce our runtime adaptation framework that allows the exchange of components in an atomic and transparent operation to provide adaptation of DBMSs concerning future changes, without hampering dependent applications.

Keywords: Database management, runtime adaptation, tailoring, components

Joint work of: Irmert, Florian; Fischer, Thomas; Lauterwald, Frank; Meyer-Wegener, Klaus

Cellular DBMS

Syed Saif ur Rahman (Universität Magdeburg, DE)

Classical Database Management Systems (DBMS) were developed decades ago and have evolved over time to handle large amount of data on high-end systems.

These DBMS are now full of layers and functionalities that are tightly coupled with there monolithic engines. Tight coupling of functionalities limits classical DBMS minimal set of functionalities as too large. Most of these functionalities are never used during life-cycles of most of the applications. Embedded and mobile systems with heterogeneous platforms, scarce resources and diversified requirements also need DBMS services for their data management needs. Classical DBMS with minimal functionalities are too large for such applications. These factors trigger the need for DBMS that is highly customizable for specialized applications.

Even for classical high-end systems, with ever increasing processing power, ever reducing processing cost and advent of high speed networks, data processing needs and DBMS complexity have increased. Complex DBMS need continuous administration and maintenance to keep system performing at optimal level resulting in high administrative and maintenance cost. Since DBMS functional components are tightly coupled with monolithic engine, tuning these systems for optimal performance is a nightmare. These systems have dozens of tuning knobs. Coupling is so tight that to assess what effect tuning one component will have on other is unpredictable. These factors trigger the need for DBMS that is tunable for required performance; however this needs transition from complex DBMS to simplified DBMS with simplified components of limited functionality and clean inter-component interaction as recommended in [1]. Considering problems of classical DBMS complexity we propose the idea of building highly customizable DBMS that will be able to self-tune it-self. In Cellular DBMS we propose constructing small to large scale DBMS through interconnected embedded DBMS (i.e. FAME-DBMS).

[1] Weikum, G., Mönkeberg, A., Hasse, C., and Zabback, P.: Self-tuning Database Technology and Information Services: from Wishful Thinking to Viable Engineering. In Proceedings of VLDB 2002.

Keywords: Embedded DBMS, Customization, Self-Tuning, Runtime-Adaptability

Joint work of: Rahman, Syed Saif ur; Rosenmüller, Siegmund, Norbert, Marko; Sunkle, Sagar; Saake, Gunter

Architecture of FAME-DBMS

Syed Saif ur Rahman (Universität Magdeburg, DE)

Embedded databases emerged to fulfill the data storage and access needs of the embedded systems. Embedded databases as per definition from [1, 2, 3] are: "Database software that is used in non-desktop systems such as cellphones, PDAs and other dedicated devices. It may be software built from the ground up for this purpose or a slimmed-down version of a larger, mainstream database management system (DBMS). For example, DB2 Everywhere is an embedded version of IBM's flagship DBMS." The term "Embedded Database" is confusing

for researchers. It depicts two meanings, one definition mentioned above gives the sense like "Database for embedded" where as the term it self can be interpreted as "Database that embed" as discussed at [4]. We believe both definitions are valid in there context and will discuss the conceived state of the art characteristics for embedded databases. As the definition specifies the fact that embedded databases are built from either ground up or are slimmed down version of large DBMS. We reject the policy for slimmed down version of large DBMS based on facts as discussed in [5, 6, 7]. We emphasize on the specialized nature of these systems and stress the need for specialized architectures and techniques for developments and operations of embedded databases. Classical architectures were developed decades ago to handle large amount of data on high-end hardware. Large DBMS that exists today have evolved over time and are now full of layers and functionalities that are tightly coupled with there monolithic engines. Slimming down these systems is compromised approach that results in embedded DBMS with compromised functionalities. FAME-DBMS is embedded DBMS based on ground-up approach with high reconfigurability while preserving the specialized characteristics for embedded databases.

[1] "Embedded database", TechEncyclopedia
[2] "Definition of: embedded database", PCMAG.COM Encyclopedia
[3] "Embedded database", THE FREE LIBRARY
[4] J. F. Koopmann, "Embedded Database Primer", dbazine.com
[5] M. Stonebraker, U. Çetintemel: "One Size Fits All": An Idea Whose Time Has Come and Gone (Abstract). ICDE 2005: 2-11
[6] M. Stonebraker, S. Madden, D. J. Abadi, S. Harizopoulos, N. Hachem, P. Helland: The End of an Architectural Era (It's Time for a Complete Rewrite). VLDB 2007: 1150-1160
[7] M. Stonebraker, C. Bear, U. Çetintemel, M. Cherniack, T. Ge, N. Hachem, S. Harizopoulos, J. Lifter, J. Rogers, S. B. Zdonik: One Size Fits All? Part 2: Benchmarking Studies. CIDR 2007: 173-184

Keywords: Customization, Re-Configurability, Embedded DBMS, Tailor-Made Data Management.

Joint work of: Rahman, Syed Saif ur; Hübner, Christian; Rosenmüller, Marko; Siegmund, Norbert; Sunkle, Sagar; Saake, Gunter

Feature-oriented refactoring of Berkeley DB

Marko Rosenmüller (Universität Magdeburg, DE)

Applications in the domain of embedded systems are heterogeneous and store an increasing amount of data. Thus, data management functionality is needed that satisfies a variety of requirements of these applications. The resource restrictions of embedded systems imply a need for tailor-made data management, i.e., data management that is customized to fulfill special requirements. In this talk we present a case study of a feature-oriented refactoring of Berkeley DB.

While existing approaches for tailoring data management solutions have significant drawbacks regarding performance and ressource consumption we show that feature-oriented programming overcomes these limitations.

Keywords: Feature-oriented programming, tailor-made data management, performance, footprint

Automated Tailoring of Infrastructure Software Product Lines

Horst Schirmeier (Technische Universität Dortmund, DE)

Especially in the resource-constrained domain of embedded systems, infrastructure software such as operating systems or libraries is being developed as a software product line (SPL). Coming with proper tool support, these systems can be configured by selecting the features needed for a specific application. However, for humans this feature-based configuration becomes increasingly complex with a growing number of configurable features, which looms in prospective SPL development. The goal of this work is to reduce this complexity. The approach is based on the observation that many configuration decisions could be automated by statically analyzing the application code on top of an infrastructure product line. This automation would reduce the configuration effort significantly. Experience gathered with our analysis tool prototype is expected to result in guidelines for future infrastructure product line APIs, further easing the analysis process.

Keywords: Infrastructure, Software Product Lines, SPL, Tailoring, Automation, Feature-based, Embedded

Joint work of: Schirmeier, Horst; Spinczyk, Olaf

Full Paper:
http://www.fame-dbms.org/publications/SPLC2007-Schirmeier.pdf

See also: H. Schirmeier and O. Spinczyk. Tailoring Infrastructure Software Product Lines by Static Application Analysis. In Proceedings of the 11th Software Product Line Conference (SPLC '07), pages 255–260. IEEE Computer, 2007.

What about your Database – Green or Fast?

Karsten Schmidt (TU Kaiserslautern, DE)

The usual performance optimization efforts for DBMSs simply address transactional throughput often by an offline analysis of the workload or by the application of limited self-tuning measures, e.g., on memory pools.

For more advanced approaches, various resources like main memory, CPU, network, and disks have to be monitored to find correlations between workload and system load in order to analyze and optimize the DBMS. New hardware having specific characteristics try to improve an upcoming "trend" – green computing. Often the processing power is intentionally reduced to save energy while the database system configuration remains untouched or is, if at all, optimized to traditional hardware environments.

Together with a light-weight online monitoring component and an energy-aware optimizer, we want to adjust our system on-the-fly to available hardware capacities, workloads, and operating goals. We will adjust our prototype system XTC to "energy efficiency" thereby testing new propagation algorithms, new logging concepts, and new memory management ideas. The integration of new hardware such as flash disks offers new opportunities to approach differing operating goals (from maximum throughput to minimum energy consumption).

Keywords: Green computing flash disk dbms

Dimensions of Variability in Embedded Systems Software

Wolfgang Schröder-Preikschat (Universität Erlangen, DE)

Design, implementation, and re-engineering of systems software, such as data base systems or operating systems, is still an ambitious undertaking. Despite, or even because, of the long history of theory and practice in this field, adapting existing systems to environments of different conditions and requirements as originally specified or assumed, in terms of functional and/or non-functional respects, is anything but simple. Especially this is true for the embedded systems domain which, on the one hand, calls for highly specialized and application-aware system abstractions and, on the other hand, cares a great deal for easily reusable implementations of these abstractions. The latter aspect becomes more and more important as embedded systems technology is faced with an innovation cycle decreasing in length. Software for embedded systems needs to be designed for variability, and this is in particular true for systems software of this domain.

The talk discusses dimensions of variability that need to be considered in the development of embedded systems software. Basing on a case study from the operating-systems field that addresses typical functional and non-functional properties of a threading subsystem, design and implementation challenges of systems software are elaborated. A selection of software engineering approaches that aid construction and maintenance of evolutionary systems software are touched on. In the discussion, problem formulation and sensitivity training come to the fore and complete solutions will take a back seat.

Keywords: Embedded systems, operating systems, reutilization, specialization, cross-cutting concerns

FAME-DBMS Challenges and Solutions

Norbert Siegmund (Universität Magdeburg, DE)

Database management systems for embedded devices have to face with several problems. Resource constraints like memory size, power consumption, and limitted processing power lead to special algorithms and architectures in this domain. For that reason, already known and implemented functionality is re-developed to meet certain constraints or different operating systems. FAME-DBMS aims to overcome these problems by creating a "Family of Embedded Database Management Systems" using the software product line approach. To implement such a DBMS, we identified special requirements and challenges by investigating current DBMS solutions in the embedded domain. Out of this domain analysis, we identified challenges for the realization of FAME-DBMS. The approaches to solve these challenges are given in the next talk.

Keywords: FAME-DBMS, Challenges, Requirements, Domain Analysis, Embedded System DBMS

Configuration of Non-Functional Properties in Software Product Lines

Julio Sincero (Universität Erlangen, DE)

The configuration of NFPs (non-functional properties) is a crucial problem in the development of software-intensive systems.

Most of the approaches currently available tackle this problem during software design. However, at this stage, NFPs cannot be properly predicted. As a solution for this problem we present the new extensions of the Feedback approach which aims at improving the configuration of NFPs in SPLs.

We introduce our set of tools that are used to support the approach.

Keywords: SPL, NFP

Product Line Technology for Infrastructure Software

Olaf Spinczyk (Technische Universität Dortmund, DE)

There are many similarities between database management systems and operating systems as well as the corresponding research communities. The aim of my introductory talk is to present the history of "tailor-made operating systems" and how the perception of this topic by the OS community changed over time. This should be understood as a motivation for database and operating system researchers to work together during this seminar and to bridge the gap to the software engineering community.

Keywords: Operating systems, database management systems, appplication-specific tailoring

Introducing a Service Oriented DBMS

Ionut Subasu (Universität Zürich, CH)

Evolving database management systems (DBMS) towards more flexibility in functionality, adaptation to changing requirements, and extensions with new or different components, is a challenging task. Although many approaches have tried to come up with a flexible architecture, there is no architectural framework that is generally applicable to provide tailor-made data management and has the capability of directly integrating existing application functionality.

We discuss an alternative database architecture that enables more lightweight systems by decomposing the functionality into services and have the service granularity drive the functionality. We propose a service-oriented DBMS architecture which provides the necessary flexibility and extensibility for general-purpose usage scenarios.

Keywords: Service, Oriented, DBMS

Joint work of: Subasu, Ionut; Dittrich, Klaus

Feature-oriented Query Processing

Sagar Sunkle (Universität Magdeburg, DE)

This is part of the FAME - DBMS project presentation. I present the query processing component proposed for FAME - DBMS. We propose to apply feature-oriented techniques to query processing by implementing feature-oriented programming atop a feature-based SQL:2003 parser. We begin by analyzing the domain of query processing and deriving feature model for query processing. We also present various benefits and challenges in such an implementation.

Keywords: Feature-oriented, Query Processing

View Integration for Context-based Data Tailoring

Letizia Tanca (Politecnico di Milano, IT)

Common to all actors in today's information world is the problem of lowering the "information noise", both in a quantitative sense – i.e., reducing the amount of data to be stored and accessed – and in a qualitative sense – i.e., enhancing the "precision" according to which the available data fit the application requirements. Thus, fitting data to the application needs is tantamount to fitting a dress to a person, and will be referred to as Data Tailoring.

The available data – possibly assembled and integrated from many, heterogeneous data sources – are tailored based on the current Context in which the application is running.

A context-guided methodology is proposed, which supports the data designer in associating, to each possible application context, an appropriate subset of the available data The methodology is composed of three basic elements: (i) a context model, capturing all the aspects – the so-called dimensions – that allow the implicit representation of the possible application contexts; (ii) a strategy for identifying, for each dimension value and independently of the others, a relevant portion of the entire data schema – the so-called partial view –; and (iii) operators for combining the partial views associated to a set of dimension values to derive the final view associated with each context.

Keywords: Context-based data tailoring, context-based data integration, view integration operators

Joint work of: Tanca, Letizia, Bolchini, Cristiana, Curino, Carlo A., Orsi, Giorgio, Quintarelli, Elisa, Rossato, Rosalba, Schreiber, Fabio A.

Staged Feature Refactoring: A Case Study of Gas Boilers Embedded Systems

Salvador Trujillo (Ikerlan Research Centre, ES)

Feature refactoring is the process of decomposing a system code into a set of features, that are a means for communicating commonalities and variabilities to stakeholders. As features encapsulate functionality, different feature compositions yield a family of different systems to form the seed of a software product line. Previous work investigated feature refactoring on the assumption that the system was designed to be extensible, this facilitated an extractive approach towards product lines.

However, software systems (embedded or otherwise) do not necessarily have a design that facilitates a feature refactoring.

Indeed, this is the case of our family of gas boiler systems. Based on our experience in feature refactoring in this case, we argue in this talk that after initial feature refactoring further action may be required to achieve a feature-based design. To attain this, this work leverages on two mechanisms: first, the granularity of refactoring is defined; and second, representations that are more abstract than code are used. This enables feature refactoring to target a coarser granularity and raise the abstraction level, respectively. Both mechanisms are intended to yield a feature-based design by means of successive stages towards feature refactoring. Our case study of a family of gas boiler systems illustrates staged feature refactoring.

Keywords: Feature refactoring, models, embedded systems