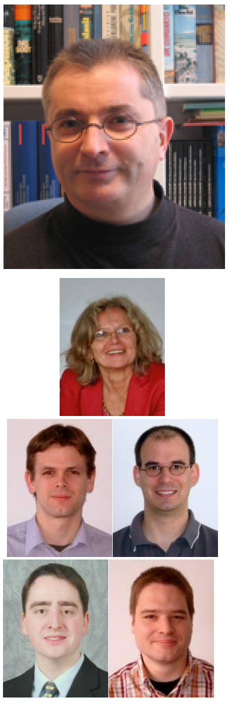
Software Construction



Staff

- Faculty:

  Univ.-Prof. Dr. rer. nat. Horst Lichter
  lichter@informatik.rwth-aachen.de

- Secretary:

  Bärbel Kronewetter
  Phone: +49 241 80 21 330
  Fax: +49 241 80 22 352

- Research Assistants:

  Dipl.-Inform. Alexander Nyßen
  (third-party funds position)

  Dipl.-Inform. Thomas von der Maßen

  Dipl.-Inform. Holger Schackmann
  (third-party funds position)

  Dipl.-Inform. Thomas Weiler
  (third-party funds position)

- Visitors:

  Dr. Shmuel Tyszberowicz, Tel Aviv University

- Student Researchers:

  Nan Mungard

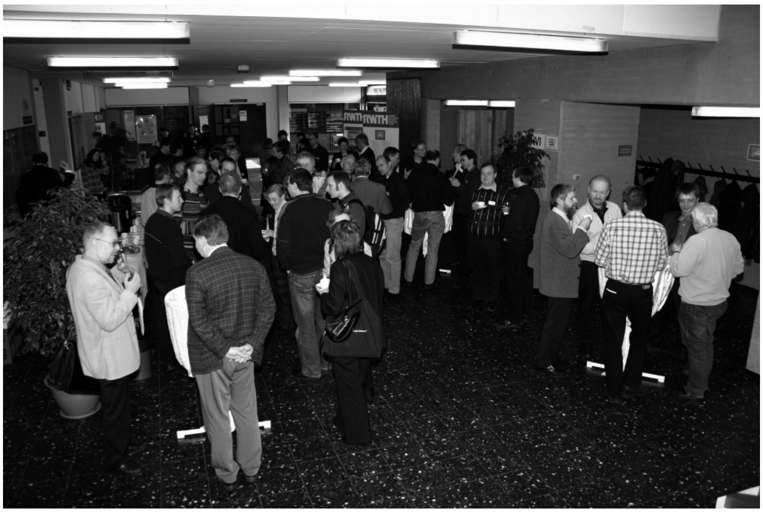
- Internet:

  Information about our research and teaching activities can be found at: http://www-lufgi3.informatik.rwth-aachen.de

Overview

Our research focuses on the development of new and advanced methods and techniques in the broad area of software construction. Currently we are running a couple of projects in the context of software product line development. Here we address especially requirements and architecture issues.

In 2005 we organized the 9th Workshop on Software Engineering Education at Universities (SEUH 2005) which is a joined workshop series of GI, SI and the German Chapter of the ACM. The workshop took place the 24th and 25th of February at the Informatics Centre. The program of the workshop offered a couple of talks as well as panel discussions and a very special session where students comment on the workshop and the presented teaching approaches. The key note was given by Prof. W. Tichy, University of Karlsruhe. Before the workshop a tutorial on a project management simulation environment (SESAM) was organized together with the software engineering chair of Stuttgart University. Altogether we could welcome about 70 software engineering professors and practitioners from Austria, Switzerland and Germany.



SEUH 2005

In spring Dr. Shmuel Tyszberowicz from Tel Aviv University visited our group. He is doing research on aspect oriented programming. Together we started a case study to investigate the benefits and drawbacks of using aspect oriented programming as an implementation technique for software product lines.

Teaching

In addition to undergraduate courses on Programming and Software Development the group offers on the graduate level the following set of courses focusing on Software Construction and Software Quality Assurance:

- Lecture *Software Quality Assurance and Project Management*
- Lecture *Product Line Development*
- Lecture *Object-Oriented Software Construction*
- Seminars on various topics
- Practical Labs

Furthermore we are responsible for the Software Engineering course of the master program Software Systems Engineering at the Thai German Graduate School of Engineering, Bangkok, Thailand. This course was given the first time in winter term 2005.

Research Projects

Requirements Engineering for Software Product Lines

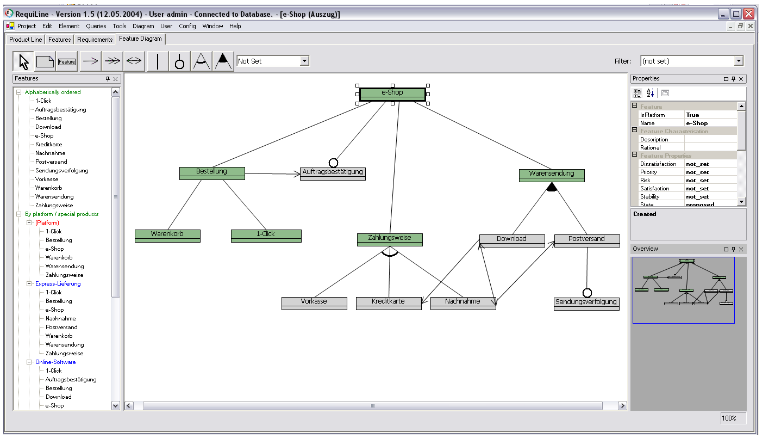
T. von der Maßen, H. Lichter

The development of a Software Product Line (SPL) is a demanding task for all stages of the software development process especially for the requirements engineering. The identification and modeling of common and variable characteristics are an essential task during the requirements engineering process. Communicating variability to stakeholders affect the success of projects significantly.

Feature based domain modeling is a well-known technique in requirements engineering of product lines to model variability within requirements. The overall objective is to model commonality and variability in a platform feature model (PLFM). From the PLFM, dedicated product feature models (PFM) can be derived. The PLFM is part of the platform requirements speicifcation and a PFM is part of a product requirements specification. Though quality models for requirements specification exists, quality models for feature models have been neglected so far. Especially it must be examined, how variability influences the claimed qualities.

Evaluating the adequacy, integrity and consistency of PLFM is of high importance. The derivation of PFMs can only be done, if the PLFM does not show any inconsistencies. Therefore the research group has defined categories of inconsistencies and identified the problems that can appear within the models and introduced metrics to determine if variability is adequately modeled.

Our research group continued furthermore the work on *RequiLine*, a requirements engineering tool that supports the management of nautural language requirements and feature models, equally. RequiLine has been enhanced by a PFM derivation wizard. This wizard guides the user through the derivation process to avoid the building of inconsistent PFMs. Furthermore a metric interface has been implemented that allows the determination of the PLFMs variation degree, to evaluate its flexibility and adequacy.


RequiLine - Version 1.5 (12.05.2004) - User admin - Connected to Database. - [e-Shop (Auszug)]

Filter: (not set)

Feature	
IsPlatform	True
Name	e-Shop
Feature Characterisation	
Description	
Rational	
Feature Properties	
Dissatisfaction	not_set
Priority	not_set
Risk	not_set
Satisfaction	not_set
Stability	not_set
State	proposed

RequiLine's feature editor

RequiLine has been evaluated in cooperation with Robert Bosch GmbH in the context of the development of motor control units. RequiLine detected several defects in the motor control units PLFM. Additionally RequiLine helped in retriving information about the variation degree and other needful statistical information, that influences the further development at Bosch.

Feature-based Architecture-Modeling for Software Product Lines

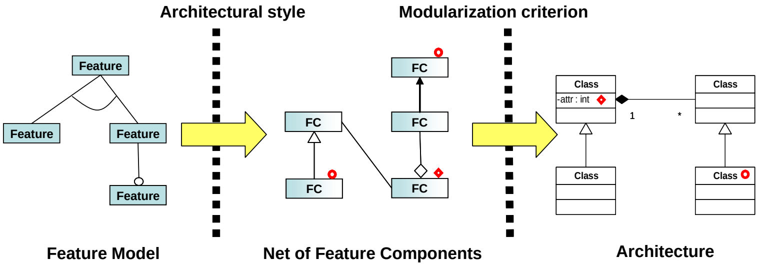
T. Weiler, H. Lichter

External cooperation: ABB Corporate Research, Ladenburg

In the last year we have improved our methodology for developing a product line platform architecture (PLPA) based on a feature model, thus allowing for smooth transition from requirements engineering to architecture modeling. While the single phases of the software development process are mostly self-contained with continuous methodology, notation and also adequate tool support, the transitions often exhibit a gap. To bridge this gap, transformations between the different methodologies and notations used in the adjacent phases are needed. But the abstraction needed for these transformations results in information loss.

By providing a methodology which minimizes the information loss between the single phases of the software engineering process, traceability of modeling decisions can be ensured. This results in a better documentation of the process and its products which in turn eases the evolution of the products and ensures return of investment.

Feature Modeling can assist this task by providing an input for the design process to identify components and structures of the PLPA. The main advantage of this procedure is that the feature model defines already a net of terms, which are identified to be crucial for the system, and the variability among them. These terms - the features - are thereby organized in a structure (here: a tree or graph) abstracting from the nebulous cloud of customers requirements. Thus, feature modeling supplements the output of the requirements process (namely the requirements specification), by providing a more formal and structured input for the following design phase. This aids the software architect finding the structure for the system and identifying architectural components. When analyzing the feature model to identify architectural component candidates, the categorization defined by the architectural style chosen has to be taken into account. This way all features modeled in the feature model are sieved by some kind of filter stack, defined by the chosen architectural style, see the following Figure.



Methodology for PLPA Modelling.

The categorization respectively modularization defined by the chosen architectural style will rarely map one to one to the features modeled in the feature model. Therefore each feature has to be analyzed to that effect which parts of its specification map into which category defined by the architectural style. We call these parts feature components (FC). We are convinced that features can not be directly mapped to architectural components like classes, aspects, or other technical components because a feature model and thus its features are an abstraction of the requirements of an SPL. Because an architectural design has to be concrete on its components, one has to analyze the features on how they contribute to architectural components. Hence this transformation, in our opinion, can not be automated, but supported by a set of transformation rules which help the designer to identify architectural components.

First, all features have to be analyzed with respect to the chosen architectural style to find how they contribute to the different categories defined by the architectural style. This way, lists of feature components are created for the different categories of the architecture. In the next step the relationships between the earlier identified feature components have to be analyzed, based on the feature descriptions and a set of transformation rules. The result of this process is a net of feature components which serves as an input for the next step. The analysis of the feature components is not straightforward, and several iteration steps are needed to obtain the final net of feature components.

During identification of the feature components each variability, which is assigned to a feature in the feature model, is analyzed to determine whether it influences one of the feature components derived from the feature. If so, the feature component concerned is also assigned the variability, which is depicted in the figure above, by a red circle (denoting an option) or red diamond (denoting an alternative). Therefore, a variability modeled in the feature model can be distributed over a set of feature components which all have the same variability with the same ID assignment. This way, traceability of the domain variability modeled in the feature model is ensured.

In the next step architectural components must be formed based on the feature components identified before, according to the modularization criteria used at the subsystem level of the architecture. In order to form e.g. classes, each component feature has to be analyzed to determine whether it denotes a class, a responsibility (method) of a class, or an attribute of a class. This way a class diagram representing the static structure of the product line platform architecture is created.

Product Line Development for Embedded Systems

A. Nyßen, H. Lichter

External cooperation: ABB Corporate Research, Ladenburg

While hardware development is largely understood and can be efficiently performed, state-of-the-art software engineering for embedded systems is far behind that of other application areas. Thus, embedded software systems are often monolithic platform-dependent systems that are built from scratch and are hard to maintain, upgrade, customize or even port to other platforms. To establish systematic development to this field is challenging, because the stringent non-functional requirements that are imposed on an embedded software system by its surrounding environment (like memory consumption or timing constraints) and the special application domains (e.g. hazardous application areas) do not allow to apply common software engineering practices ”as is” but require that they have to be ”tailored” to meet these new demanding requirements. Inspired by the large success product line engineering has brought to hardware development, it seems to be a promising approach to gain more reuse, higher product quality and lower product development costs in software development also, especially in the development of embedded software systems.

However, product-line practices cannot be reasonably applied if no systematic development is established in the developing organization, which can be taken as a firm basis to build upon. That is why past work of this project focussed on methodical aspects and - furthermore inspired by the practical needs of our business coorporation partners - resulted in the definition of an iterative development process and a detailed design method (at first for the single product case) that are capable of forming the basis for the application of more far-reaching product line engineering practices.

While the introduction and optimization of the conceived process and methodology with our business cooperation partners has been considerably supported, current research work is this project is focussing on a more sophisticated UML2-based notation for product-line architectural models, as well as on methology to support the design of those. This elevated concentration is done, because a product-line architecture is being regarded as the major core asset of a product line - as it forms the bases for all succeeding development activities inside the product-line life cycle. In this context, much work is also spent on the development of a visual modeling tool supporting the developed notation and design methodology, called ViPER (Visual Modeling Platform for Embedded System ARchitectures). It is intended to be a research prototype to demonstrate the conceived notation and methodology. Furthermore it will later be employed to demonstrate the integration of architectural models with those models of preceeding and succeeding development phases, e.g. by offering source code generation capabilities based to support the implementation.

Process and Tool Support for the Maintenance of Hierarchical Product Lines

H. Schackmann, H. Lichter

External cooperation: Kisters AG, Aachen

The parallel development and maintenance of multiple customer specific products within a product line requires serious efforts for coordination and monitoring. This is especially the case, when different product lines are based on a set of common assets and must share the development resources. The particular products may have different release plans that must be fulfilled. Development resources must be shared efficiently between product development projects and platform development. The common platform, as well as the platform of each product line based on it, must serve the sometimes diverging needs of the products. But it must be prevented that this results in different variants of a platform that are maintained in parallel. Under these circumstances adequate processes with suitable tool support are necessary to take advantage of the synergies in product line development.

In the last year we designed a development process in cooperation with Kisters AG that is targeted to fit the needs of evolving product lines better than prevalent project-centric approaches. On this basis there arise many questions for the detailed sub processes which are subject of ongoing work. Amongst others we developed guidelines for the requirements management process and support for risk management. Current work is focused on putting these measures into practice, which will include the customizing of the processes to the needs of different product lines.

Driven by the practical experiences we are furthermore working on concepts for continuous life-cycle scoping for software product lines. We identified the necessity of investigating the cost effects of product line variabilities during the evolution of the product line. This requires a suitable approach for costing which differs from project-centric cost accounting. The intention is to use this information to enable substantiated decisions to control the evolution of variation points in a software product line. An improved transparency of the different activities, their progress and their costs will support the product-line wide coordination and planning.

Other Activities

Board Member of the *GI-Fachgruppe 2.1.6. Requirements Engineering*, *H. Lichter*

Member of the program comittee, GI-Workshop Informatik LIVE für Users LIFE flexible Integration machts möglich!, Bonn, 19.9.2005

Member of the program comittee, GI-Conference *Modellierung 2004*, Marburg, 24.-26.3.2004, *H. Lichter*

Member of the program comittee, GI-Workshop *SEUH 2005*, Aachen, 24.-25.2.2005, *H. Lichter*

Organization of SEUH 2005 and SESAM Tutorial, RWTH Aachen, 23.-25.2.2005, *H. Lichter*

Reviewer for dpunkt-Verlag Heidelberg and computing reviews, *H. Lichter*

Organization of the Computer Science Department's mentors program, *H. Lichter*

Member of the Computer Science Department's commitee for Service-Lehre, *H. Lichter*

Member of the examination board of Computational Material Science, *H. Lichter*

Organization of the Beginner's Course in Computer Science 2005, *H. Lichter, T. Weiler*

Speaker of the GI-Arbeitskreis *Werkzeuge für die Produktlinienentwicklung* of the GI-Fachgruppe Requirements Engineering, *T. von der Maßen*

Talks and Publications

Talks

H. Lichter: *Techniques, Processes and Methods for Requirements Engineering,* ABB Corporate Research Centre, Ladenburg

H. Lichter: *Evolution of Software Product Lines,* University Karlsruhe

H. Lichter: *Grundlagen des Requirements Engineering,* Kisters AG, Aachen

T. von der Maßen: *Determining the Variation Degree of Feature Models, in Software Product Lines,* 9th International Conference, SPLC 2005, Rennes, France

T. Weiler: *Are Aspects useful for Managing Variability in Software Product Lines? A Case Study,* An Early Aspects Workshop at at the 9th International Conference, SPLC 2005, Rennes, France

Publications

Thomas von der Maßen, Klaus Müller, John MacGregor, Eva Geisberger, Jörg Dörr, Frank Houdek, Harbhajan Singh, Holger Wußmann, Hans-Veit Bacher, Barbara Paech: *Einsatz von Features im Software-Entwicklungsprozess,* Abschlussbericht des GI-Arbeitskreises *Features*, Aachener Informatik Berichte AIB-2005-18, 2005

Alexander Nyßen, Horst Lichter, Jan Suchotzki, Peter Müller, Andreas Stelter: *UML2-basierte Architekturmodellierung kleiner eingebetteter System - Erfahrungen einer Feldstudie,* In Klein, Rumpe, Schätz (Hrsg.), Tagungsband des Dagstuhl-Workshops Modellbasierte Entwicklung eingebetteter Systeme (MBEES), Technischer Bericht, TU Braunschweig, TUBS-SSE-2005-01, 2005

Klaus-Peter Löhr, Horst Lichter: *Software Engineering im Unterricht der Hochschulen - SEUH 9,* dpunkt-verlag, ISBN 3-89864-328-X, 2005

Alexander Nyßen, Shmuel Tyszberowicz, Thomas Weiler: *Are Aspects useful for Managing Variability in Software Product Lines? A Case Study,* Aspects and Software Product Lines: An Early Aspects Workshop at SPLC-Europe 2005, Rennes (France), September 2005

Thomas von der Maßen, Horst Lichter: *Determining the Variation Degree of Feature Models,* In Software Product Lines: 9th International Conference, SPLC 2005, LNCS, Volume 3714, Oct 2005, pp. 82 88