MoVES Newsletter

Work Package 7 - Incremental design and verification
University in Focus: University of Namur

MoVES Newsletter, No. 2, September 2010

Editorial

Dear Reader,

This is the second issue of the MoVES newsletter. MoVES stands for "Modelling, Verification and Evolution of Software" and addresses these fundamental issues in software engineering. The project is sponsored by the Belgian government (belspo IAP programme). Each issue of the newsletter presents a partner and a work package.

This issue presents results and ongoing research of *Work Package 7-incremental design and verification*. The idea of WP7 is to bridge the main topics of the project. Evolution poses a challenge for both modelling and verification: We want to reuse the modelling and verification effort as much as possible when evolving a software system. This requires innovative structuring of the software and the verification process. This issues also presents the *PReCISE Research Centre* of the *University of Namur (FUNDP)*, and shows the variety of research pursued there.

Enjoy reading!

Pierre-Yves Schobbens, Nicolas Genon and Andreas Classen

Upcoming Events & Recent Joint Publications

- 25th IEEE/ACM International Conference on Automated Software Engineering, **ASE 2010**, 20-24 September 2010, Antwerp Belgium.
  - **IWPSE-Evol 2010**: 4th International Joint ERCIM/IWPSE Symposium on Software Evolution, 20 September.
  - **Moves-Verif**, 21 September.
  - **WASDeTT-3**: 3rd International Workshop on Academic Software Development Tools, 20 September.
- ACM/IEEE 13th International Conference on Model Driven Engineering Languages and Systems, **MODELS 2010**, 3-8 October 2010, Oslo Norway.
  - **ME 2010**: International Workshop on Models and Evolution, 3 October.
  - **ACES-MB 2010**: 3rd International Workshop on Model Based Architecting and Construction of Embedded Systems, 4 October.
- 9th BElgian-NEtherlands software eVOLution seminar, **BENEVOL 2010**, 16-17 December 2010, Lille France.
- *Joint publications to appear* in Volume 27(4), July-August, of **IEEE Software** devoted to *Software Evolution*, with guest editors T. Mens, Y.-G. Guéhéneuc, J. Fernández-Ramil, and M. D'Hondt, with two MoVES contributions:
  - *Reverse Engineering on the Mainframe: Lessons Learned from "In Vivo" Research*, J. van Geet and S. Demeyer.
  - *A Lightweight Sanity Check for Implemented Architectures*, A. van Deursen and E. Bouwers.
- *Submission deadlines:*
  - *October 15th 2010* – 15th European Conference on Software Maintenance and Reengineering, **CSMR 2011**.
  - *October 31st 2010* – Special Issue on **"Automated Software Evolution"** of the Elsevier Journal on Systems and Software.
  - *November 10th, 2010* – 5th International Workshop on Variability Modelling of Software-intensive Systems, **VaMoS 2011**, January 27th - 29th 2011, Namur Belgium.
  - *December 22nd, 2010* – 1st International Workshop on Variability-intensive Systems Testing, Validation and Verification, **VAST 2011**, March 21-25, 2011 - Berlin Germany.
  - *February 14th, 2011* – 19th IEEE International Requirements Engineering Conference, **RE 2011**, August 29th - September 2nd 2011, Trento Italy. The theme will be **"Requirements in Motion"**.

Andreas Classen, Patrick Heymans, Axel Legay, Jean-François Raskin and Pierre-Yves Schobbens

Model Checking Software Product Lines

In product line engineering, systems are developed in *families* and differences between family members are expressed in terms of *features*. Formal modelling and verification is an important issue in this context as more and more critical systems are developed this way. Since the number of systems in a family can be exponential in the number of features, two major challenges are the scalable modelling and the efficient verification of system behaviour. Most current proposals addressing these challenges suffer from two main limitations. Their models often fail to recognise the importance of features as a unit of difference and none of the proposals provides concrete means for checking behavioural models against temporal properties.

We tackle these challenges at a *foundational* level. We propose FTS, *Featured Transition Systems*, a formalism designed to describe the combined behaviour of an entire system family. FTS are transition systems in which transitions are labelled with features of the product line (in addition to being labelled with actions). The semantics of FTS is parameterised and allows to obtain the behaviour of each family member. The principal advantages of FTS over existing work are (i) the modelling of variability as a first-class citizen, (ii) the ability to reason about the whole product line, (iii) to model very detailed behavioural variations, and (iv) to take feature dependencies and incompatibilities into account.

FTS come with a tool-supported model checking approach that allows to verify FTS against linear temporal logic (LTL) and branching temporal logic (CTL) properties. The purpose of the approach is to verify all members of a family at once and to pinpoint those that violate a property. An empirical evaluation conducted for the LTL algorithms showed substantial gains over individual member verification. For LTL model checking, we developed a tool using as input language a variant of Promela, in which statements can be guarded by features. For CTL model checking, we extended the NuSMV model checker and use an existing feature-oriented extension of the NuSMV language as input language. Both tools are available at the FTS website.

This was in part published as *Model Checking Lots of Systems: Efficient Verification of Temporal Properties in Software Product Lines*, in *32nd International Conference on Software Engineering, ICSE 2010, May 2-8, 2010, Cape Town, South Africa, Proceedings, pp. 335-344.*

Further Information: `http://www.info.fundp.ac.be/~acs/fts/`
Contact: *acs@info.fundp.ac.be*

Peter Ebraert, Andreas Classen, Patrick Heymans and Theo D'Hondt

Feature Diagrams for Change-Oriented Programming

Several methods for implementing variability in *software product line engineering* exist. Traditional approaches sometimes lack modularity and reusability or require a significant amount of manual labour. *Feature-oriented programming* overcomes these problems by using *features* as first-class abstractions during the development process. A feature closely maps to a variant of a variable requirement and variability can be implemented merely by selecting or deselecting features. In *Change-Oriented Programming* (ChOP), a specific approach to feature-oriented programming, each feature is represented by a set of *changes* applied to a base system. Feature composition, in this case, becomes change composition. One of the challenges in ChOP is to make sure that a composition of several features, viz. changes, is free of harmful interactions: two or more changes that must be applied together for a composition to be valid. However, there is currently no formal framework for ChOP, that would allow to define this kind of property and serve as a reference for ChOP implementations.

We fill this gap with two contributions. First, we formalise the concepts and properties of ChOP using common set theory. Furthermore, we define properties such as *composability* of a set of changes, that need to be checked before they can be composed. We then show that ChEOPS, the proof-of-concept implementation of ChOP, adheres to this model. An interesting observation is that the model is quite close to *feature diagrams*, a software product line engineering notation used to model the variability and manage the interactions of an application. The main purpose of feature diagrams is to model which combinations of features are allowed and disallowed in a software product line. Our second contribution is to map the formal model of ChOP to feature diagrams, whereby we are able to reuse their well-studied semantics as well as existing analysis tools. We prove that *composability* of features in ChOP is equivalent to checking whether the product they form satisfies the feature diagram.

These results are published as *Feature Diagrams for Change-Oriented Programming*, in *Proceedings of the 10th International Conference on Feature Interactions in Software and Communication Systems (ICFI'09) held in conjunction with Distributed Computing Techniques 2009 (DisCoTec'09), Lisbon, Portugal, pp. 107-122, June 2009.*

Further Information: `http://peter.ebraert.be`
Contact: *peter.ebraert@ua.ac.be*

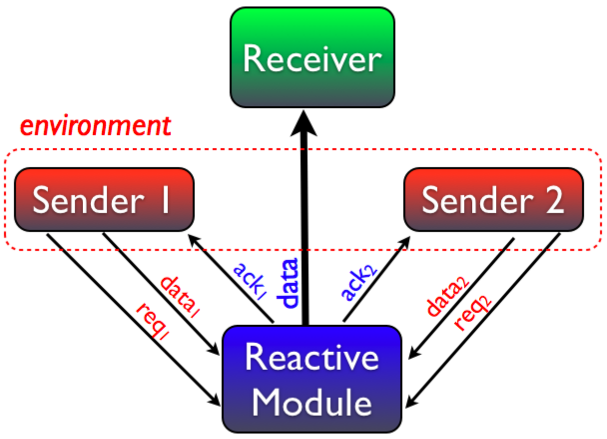
Emmanuel Filiot, Naiyong Jin and Jean-François Raskin

Incremental Synthesis from Linear Temporal Specifications

A *reactive module* is a system that interacts with an environment. At each time point, it receives some input signals from the environment and produces output signals. As in general there is no assumption on the environment behavior, the input signals are *uncontrollable*. The figure shows some reactive module that interacts with two senders. The senders may request the module to transmit some data to a receiver, and the module may acknowledge the request. It can also transmit the input data to a receiver.

Given a reactive module and a specification of desired properties on its execution, the *model-checking* problem asks to verify that this property is indeed satisfied by the module. More ambitious is the *synthesis* problem which asks to automatically generate a reactive module that satisfies a given specification. The *linear-time temporal logic (LTL)* is well-suited to describe temporal properties of reactive modules, and has been first studied in the context of synthesis by Pnueli and Rosner in 1989. For instance, in the system below, one may want to synthesize a module which satisfies the following properties (which can be easily expressed in LTL): *it is always the case that when there is a request, it is eventually acknowledged* (liveness property), and *it is always the case that two requests are not acknowledged at the same time* (safety property).

While the LTL synthesis problem is 2-ExpTime-Complete, there have been several recent progresses that let us hope to solve it in practice. In particular, we have proposed an approach based on a reduction to games played on graphs. Viewing the synthesis problem as a game is natural: as the environment behavior is uncontrollable, it has to be considered as adversarial, and the reactive system should be able to ensure the specification no matter what the environment is doing. Therefore a correct reactive system is nothing else then a winning strategy in a game between the system and its environment. Specifically, we have shown that the LTL synthesis problem can be reduced to solving *safety games*, in which the system wants to avoid some bad game positions.



Safety games are simple and support compositional reasoning. Furthermore, we have shown that the games that we need to consider have structural properties that allows us to compactly represent sets of winning positions using antichains. The new algorithms that we have proposed are compositional and incremental. While even small LTL formulas where out of reach of previous approaches, we can now synthesize modules from specifications which are *several pages* long.

Further Information: `http://www.antichains.be/acacia/`
Contact: *efiliot@ulb.ac.be, naiyjin@ulb.ac.be, jraskin@ulb.ac.be*

Benoît Vanderose, Flora Kamseu and Naji Habra

Towards a Model-Centric Quality Assessment

A vast amount of software measurement methods have been defined for quality assessment in Software Engineering. However, the apparition of new paradigms (e.g., model-driven, software product lines, etc.) coupled with this spread of measurement methods increase the occurrences of misguided and inefficient use. Measurement thus remains a control activity and fails to appeal to developers as a support to guide them.

This research work introduces a Model-Centric Quality Assessment (MoCQA) framework relying on a quality metamodel and supporting a flexible integration of quantitative quality assessment all along the software development lifecycle. This framework provides a top-down two-level methodology.

The first level (conceptual) focuses on the generation of a customised assessment quality model (CAQM) for the software project. The underlying quality metamodel ensures the robustness and coherence of the model generation and evolution. The second level (operational) implements a concrete measurement plan using the generated CAQM for a systematic and/or semi-automated analysis of the measurement values. It also allows different backtracking between those steps (and substeps) to support the CAQM 'lifecycle' which is one of the main strength of the framework.

So far, early results highlight the positive impact of the possibility to refine and let evolve the CAQM as well as the flexibility allowed by the quality metamodel.

Further Information: `http://www.fundp.ac.be/universite/personnes/page_view/01006212/`
Contact: *benoit.vanderose@fundp.ac.be, flora.kamseu@fundp.ac.be, naji.habra@fundp.ac.be*

Raimundas Matulevičius, Naji Habra and Flora Kamseu

Validity of the Documentation Availability Model: Experimental Definition of Quality Interpretation

System and software documentation is a necessity when making selection and acquisition decisions, when developing new and/or improving existing system and software functionality. A proper documentation becomes even more crucial for open source systems (OSS), where, typically, stakeholders from different communities are involved. However there exist only limited or no methodology to assess documentation quality. In this paper we present a quality model and a comprehensive method to assess quality of the OSS documentation availability (DA). Our contribution is threefold. Firstly, based on the criteria defined by Kitchenham et al. we illustrate the theoretical validity of the DA model. Secondly, we execute the first step towards the empirical validity of the DA model. Finally, our work results in a comprehensive and empirically grounded interpretation model of the documentation quality.

Further Information: `http://www.springerlink.com/content/c6n073v9t5068551/`
Contact: *rma@info.fundp.ac.be, nha@info.fundp.ac.be, fka@info.fundp.ac.be*

Fabian Gilson and Vincent Englebert

A Transformational Approach for Component-Based Distributed Architectures

Nowadays information systems are becoming larger, more complex and they are frequently integrating *Commercial Off-The-Shelf* components. Furthermore, stakeholders are having architecturally significant non-functional requirements that may influence the system architecture at early architecture design stages. As systems are intended to cope with evolving requirements, traceability mechanisms are needed to keep track of design decisions and resulting system architectures. In addition, distributed systems are often deployed on heterogeneous physical infrastructures with many deployment constraints. The present work aims at defining a transformation-oriented design method mixing architecture and requirement models in order to define, build, abstractly deploy and transform component-based information systems.

Further Information: `http://www.info.fundp.ac.be/~fgi/`
Contact: *fgi@info.fundp.ac.be, ven@info.fundp.ac.be*

Jean-Noël Colin

Digital Rights Management for Sensitive Data

A new research project, funded by FSR (FUNDP's *Fonds Spécial Recherche*) is being started, that aims at defining a model for the digital rights management applied to sensitive data exchanged between parties, taking stock of approaches already proposed in the field of commercial distribution of digital content, for instance multimedia content.

The interest for traveling data is motivated by the always greater ubiquitous character of users and data. Information about individuals is exchanged, with or without prior knowledge and consent of the concerned person, which can cause privacy issues. It is the case for instance in social networks, in the medical sector or in public administrations. There is thus a real need to define access control mechanisms based on actual access rules, that can interoperate in heterogeneous technical environment.

Important amount of work has already been published around authorization and access control models, aiming at an improved match with organizational needs, but also with legal and regulatory requirements. The objective of the project is thus to combine access control models, expression mechanisms for digital rights, in an interoperable way, allowing automatic processing, as well as enforcement mechanisms into a single approach.

The project will focus on the case of the Electronic Medical Record (EMR); eHealth is definitely an area where ICT is becoming more and more pervasive, but also where the constraints that govern access to information are very complex.

Institutions active in the field (IT department of hospitals, network of hospitals) are collaborating to the project to guarantee that the proposed approach is appropriate and realistic.

Further Information: `http://www.info.fundp.ac.be/~jnc/`
Contact: *jean-noel.colin@fundp.ac.be*

Jean-Noël Colin

Innovative Technologies for an Engaging Classroom

iTEC is a FP7 project, under the theme of Technology-enhanced learning, that starts in September 2010 for a 48 months period.

It is a large-scale pilot involving up to 1,000 classrooms focused on Learning in the 21st Century and the design of the future classroom. Partners include 15 Ministries of Education, leading ICT vendors, innovative SMEs, TEL researchers, teacher educators and experts in school validations and pedagogical evaluation. The key aim is to develop engaging scenarios for learning in the future classroom that can be validated in large-scale pilots and subsequently taken to scale.

iTEC produces meaningful pedagogical scenarios (assisted by semantic web technology) for the future classroom and, from these, derives learning activities and new approaches to assessment that engage teachers, learners and stakeholders outside the school. These are then tested and evaluated in the largest pan-European validation with schools yet undertaken. The iTEC technology approach will make the technical components, (people, tools, services and content) required by the scenarios, interoperable and discoverable, so that teachers can more easily select and combine relevant components tailored to the future classroom scenario of their choice. This is in line with current trends in which teachers can choose from a wide variety of loosely coupled tools and where interactive whiteboards and other interactive, multi-touch technologies may be acting as a 'gateway' for teachers to start exploring the further use of digital technologies in their classrooms.

Combined with this, iTEC will research the skills and competences needed by teachers in the future classroom and equip teachers, both within and beyond the project, with the pedagogical knowledge and skills needed to implement project scenarios. Having identified scenarios with the maximum potential to have a transformative effect on the design of the future classroom, the project will implement a mainstreaming strategy designed to ensure that work carried out in the large-scale pilots contribute to the educational reform process.

Our role in the project is in the design and development of the iTEC shell, i.e. the container that hosts all different components supporting the learning activities, and in particular in all the aspects related to identity and access management throughout the project

Further Information: `http://www.info.fundp.ac.be/~jnc/`
Contact: *jean-noel.colin@fundp.ac.be*

Amanuel Koshima, Vincent Englebert and Philippe Thiran

Distributed Collaborative Model Editing Framework (DiCoMEF)

In today's complex business world, industries need to develop complex enterprise scale software applications in order to survive in business and extend their market shares. In an effort to overcome this complexity, a Model Driven Engineering approach has been adopted by the Software Engineering community. Specifically, Domain Specific Modeling Languages (DSML) are commonly used to specify the structure, behavior and requirements of business applications using domain concepts rather than computer technology. Even though DSMLs are matured over time and frequently used by industries, few tool support is available to manage models. In the context of collaborative work, industries adopt a central repository with locking and merging techniques to ensure collaboration.

In this work, we proposed a Distributed Collaborative Model Editing Framework (DiCoMEF) where every developer has his/her own local copy of the global specification models and meta-models. DiCoMEF handles change (modification) propagation and change management to ensure collaboration among developers. Developers use message exchanges to propagate their local modifications to other developers. Specifically, a set of sequences of operations that may concern both models and their meta-models are used for message exchange. These sequences of operations are captured and specified in a history meta-model. As part of change management, a controller is responsible to supervise modifications of models and meta-models. This framework allows every developer to modify his/her local copy and propagate his/her local updates to the controller. The controller is responsible to propagate changes, to detect conflicts and to reconcile them under the supervision of the developer. Updates are then propagated to all developers.

Further Information: `http://www.info.fundp.ac.be/`
Contact: *amanuel.koshima@fundp.ac.be, vincent.englebert@fundp.ac.be, philippe.thiran@fundp.ac.be*

Partners

Universiteit Antwerpen

FUNDP NAMUR

UMH

MoVES Modelling, Verification and Evolution of Software

Vrije Universiteit Brussel

KATHOLIEKE UNIVERSITEIT LEUVEN

ULg UNIVERSITÉ de Liège

ULB

INRIA centre de recherche LILLE - NORD EUROPE

Imperial College London

TU Delft Delft University of Technology