IN PARTNERSHIP WITH:
Institut national des sciences appliquées de Rennes

Université Rennes 1

Activity Report 2014

Project-Team DIVERSE

Diversity-centric Software Engineering

IN COLLABORATION WITH: Institut de recherche en informatique et systèmes aléatoires (IRISA)

RESEARCH CENTER
Rennes - Bretagne-Atlantique

THEME
Distributed programming and software engineering

Table of contents

Project-Team DIVERSE

Keywords: Model-driven Engineering, Adaptive Systems, Meta-modeling, Service Oriented Architecture, Software Engineering

Creation of the Project-Team: 2014 July 01.

1. Members

Research Scientists

Benoit Baudry [Team leader, Inria, Researcher, HdR]
Benoit Combemale [Inria, Researcher]

Faculty Members

Mathieu Acher [Univ. Rennes I, Associate Professor]
Olivier Barais [Univ. Rennes I, Associate Professor, HdR]
Arnaud Blouin [INSA Rennes, Associate Professor]
Johann Bourcier [Univ. Rennes I, Associate Professor]
Jean-Marc Jezequel [Univ. Rennes I, Professor, HdR]
Noël Plouzeau [Univ. Rennes I, Associate Professor]

Engineers

Didier Vojtisek [Inria, Senior Engineer]
Fabien Coulon [Inria]
Jean-Emile Dartois [Inria, until Oct 2014]
Erwan Daubert [Univ. Rennes I, until Apr 2014]
André Elie [Inria]
Jean Parpaillon [Inria, from Dec 2014]
Nicolas Sannier [Inria, from Feb 2014 until Jul 2014]
François Tanguy [Inria, from Apr 2014]
Maxime Tricoire [Inria]

PhD Students

Jose Angel Galindo Duarte [Inria]
David Mendez Acuna [Inria]
Johan Pelay [Inst. de Recherche Technologique B-COM, from Oct 2014]
Marcelino Rodriguez Cancio [Univ. Rennes 1, from Jun 2014]
Julien Richard-Foy [Zenexity, CIFRE]
Francisco-Javier Acosta Padilla [Univ. Rennes I]
Sana Ben Nasr [Inria]
Jacky Bourgeois [Co-tutelle Univ. Rennes 1 with Open University]
Mohamed Boussaa [Inria]
Erwan Bousse [Univ. Rennes I]
Guillaume Bécan [Univ. Rennes I]
Kevin Corre [Orange Labs, from Oct 2014]
Thomas Degueule [Inria]
Joao Ferreira Filho [Inria]
Inti Gonzalez Herrera [Univ. Rennes I]
Pierre Laperdrix [Univ. Rennes I, from Oct 2014]
Valeria Lelli Leitao Dantas [Inria]
Ivan Paez Anaya [Univ. Rennes I]
Hamza Samih [All4Tec, CIFRE]
Kwaku Yeboah-Antwi [Inria]

Post-Doctoral Fellows

Walter Rudametkin Ivey [Inria, until Aug 2014]
Simon Allier [Inria]
Marco Biazzini [Inria, until May 2014]
Edward Mauricio Alferez Salinas [Inria, until Nov 2014]

Visiting Scientists

Dionysios Efstathiou [Univ. Rennes I, from Mar 2014 until May 2014]
Kahina Hamadache [Univ. Rennes I, from Sep 2014 until Oct 2014]
Aymeric Hervieu [Ouest Valorisation, from Feb 2014]
Vivek Nallur [Inria, Mar 2014]

Administrative Assistant

Loic Lesage [Inria]

Others

Victor Vranceanu [Inria, Intern, from Mar 2014 until Jul 2014]
Geoffrey Alexandre [Inria, Intern, from May 2014 until Aug 2014]
Mathias Billoir [Inria, Intern, from Jun 2014 until Sep 2014]
Louis-Marie Guillemot [Inria, Intern, from Jun 2014 until Sep 2014]
Jiyoung Kim [Inria, Intern, from Jun 2014 until Sep 2014]
Eric Manzi [Inria, Intern, from Jun 2014 until Aug 2014]
Chengcheng Pu [Inria, Intern, from Mar 2014 until Aug 2014]
Olivier Beaudoux [External Collaborators, Associate Professor, ESEO, Angers, France]
Robert France [Inria, Advanced Research position, from Sep 2014]

2. Overall Objectives

2.1. Overall objectives

DIVERSE's research agenda is in the area of software engineering. In this broad domain we develop models, methodologies and theories to address the challenges raised by the emergence of several forms of diversity in the design, deployment and evolution of software-intensive systems. The emergence of software diversity is an essential phenomenon in all application domains that we investigate with our industrial partners. These application domains range from complex systems such as systems of systems (in collaboration with Thales and DGA) and Instrumentation and Control (with EDF) to pervasive combinations of Internet of Things and Internet of Services (with TellU and Software AG) and tactical information systems (with the firefighter department). Even if today these systems are apparently radically different, we envision a strong convergence of the scientific principles underpinning their construction and validation towards **flexible and open yet dependable systems**. In particular, we see that the required flexibility and openness raise challenges for the software layer of these systems that must deal with four dimensions of diversity: the **diversity of languages** used by the stakeholders involved in the construction of these systems; the **diversity of features** required by the different customers; the **diversity of runtime environments** in which software has to run and adapt; the **diversity of implementations** that are necessary for resilience through redundancy.

In this context, the major software engineering challenge consists in handling **diversity** from variability in requirements and design to heterogeneous and dynamic execution environments. In particular this requires considering that the software system must adapt, in unpredictable ways, to changes in the requirements and environment. Conversely, explicitly handling of diversity is a great opportunity to allow software to spontaneously explore alternative design solutions. Concretely, we want to provide software engineers with the ability:

- to characterize an 'envelope' of possible variations
- to compose 'envelopes' (to discover new macro envelopes in an opportunistic manner)
- to dynamically synthesize software inside a given envelop

The major scientific objective that we must achieve to provide such mechanisms for software engineering is synthesized below

Scientific objective for DIVERSE: Automatically **compose and synthesize software diversity** from design to runtime to **address unpredictable evolutions of software-intensive systems**

Software product lines and associated variability modeling formalisms represent an essential aspect of software diversity, which we already explored in the past and that represent a major foundation of DIVERSE's research agenda. However, DIVERSE also exploits other foundations to handle new forms of diversity: type theory and models of computation for the composition of languages; distributed algorithms and pervasive computation to handle the diversity of execution platforms; functional and qualitative randomized transformations to synthesize diversity for robust systems.

3. Research Program

3.1. Scientific background

3.1.1. Model-driven engineering

Model-Driven Engineering (MDE) aims at reducing the accidental complexity associated with developing complex software-intensive systems (e.g., use of abstractions of the problem space rather than abstractions of the solution space) [148]. It provides DIVERSE with solid foundations to specify, analyze and reason about the different forms of diversity that occur through the development lifecycle. A primary source of accidental complexity is the wide gap between the concepts used by domain experts and the low-level abstractions provided by general-purpose programming languages [116]. MDE approaches address this problem through modeling techniques that support separation of concerns and automated generation of major system artifacts from models (*e.g.,* test cases, implementations, deployment and configuration scripts). In MDE, a model describes an aspect of a system and is typically created or derived for specific development purposes [94]. Separation of concerns is supported through the use of different modeling languages, each providing constructs based on abstractions that are specific to an aspect of a system. MDE technologies also provide support for manipulating models, for example, support for querying, slicing, transforming, merging, and analyzing (including executing) models. Modeling languages are thus at the core of MDE, which participates to the development of a sound *Software Language Engineering*[1], including an unified typing theory that integrate models as first class entities [151].

Incorporating domain-specific concepts and high-quality development experience into MDE technologies can significantly improve developer productivity and system quality. Since the late nineties, this realization has led to work on MDE language workbenches that support the development of domain-specific modeling languages (DSMLs) and associated tools (*e.g.,* model editors and code generators). A DSML provides a bridge between the field in which domain experts work and the implementation (programming) field. Domains in which DSMLs have been developed and used include, among others, automotive, avionics, and the emerging cyber-physical systems. A study performed by Hutchinson et al. [123] provides some indications that DSMLs can pave the way for wider industrial adoption of MDE.

More recently, the emergence of new classes of systems that are complex and operate in heterogeneous and rapidly changing environments raises new challenges for the software engineering community. These systems must be adaptable, flexible, reconfigurable and, increasingly, self-managing. Such characteristics make systems more prone to failure when running and thus the development and study of appropriate mechanisms for continuous design and run-time validation and monitoring are needed. In the MDE community, research is focused primarily on using models at design, implementation, and deployment stages of development. This work has been highly productive, with several techniques now entering a commercialization phase. As software systems are becoming more and more dynamic, the use of model-driven techniques for validating and monitoring run-time behavior is extremely promising [131].

[1] See http://planet-sl.org

3.1.2. Variability modeling

While the basic vision underlying *Software Product Lines* (SPL) can probably be traced back to David Parnas seminal article [141] on the Design and Development of Program Families, it is only quite recently that SPLs are emerging as a paradigm shift towards modeling and developing software system families rather than individual systems [139]. SPL engineering embraces the ideas of mass customization and software reuse. It focuses on the means of efficiently producing and maintaining multiple related software products, exploiting what they have in common and managing what varies among them.

Several definitions of the *software product line* concept can be found in the research literature. Clements *et al.* define it as *a set of software-intensive systems sharing a common, managed set of features that satisfy the specific needs of a particular market segment or mission and are developed from a common set of core assets in a prescribed way* [138]. Bosch provides a different definition [103]: *A SPL consists of a product line architecture and a set of reusable components designed for incorporation into the product line architecture. In addition, the PL consists of the software products developed using the mentioned reusable assets*. In spite of the similarities, these definitions provide different perspectives of the concept: *market-driven*, as seen by Clements *et al.*, and *technology-oriented* for Bosch.

SPL engineering is a process focusing on capturing the *commonalities* (assumptions true for each family member) and *variability* (assumptions about how individual family members differ) between several software products [110]. Instead of describing a single software system, a SPL model describes a set of products in the same domain. This is accomplished by distinguishing between elements common to all SPL members, and those that may vary from one product to another. Reuse of core assets, which form the basis of the product line, is key to productivity and quality gains. These core assets extend beyond simple code reuse and may include the architecture, software components, domain models, requirements statements, documentation, test plans or test cases.

The SPL engineering process consists of two major steps:

1. **Domain Engineering**, or *development for reuse*, focuses on core assets development.
2. **Application Engineering**, or *development with reuse*, addresses the development of the final products using core assets and following customer requirements.

Central to both processes is the management of **variability** across the product line [118]. In common language use, the term *variability* refers to *the ability or the tendency to change*. Variability management is thus seen as the key feature that distinguishes SPL engineering from other software development approaches [104]. Variability management is thus growingly seen as the cornerstone of SPL development, covering the entire development life cycle, from requirements elicitation [153] to product derivation [158] to product testing [137], [136].

Halmans *et al.* [118] distinguish between *essential* and *technical* variability, especially at requirements level. Essential variability corresponds to the customer's viewpoint, defining what to implement, while technical variability relates to product family engineering, defining how to implement it. A classification based on the dimensions of variability is proposed by Pohl *et al.* [143]: beyond **variability in time** (existence of different versions of an artifact that are valid at different times) and **variability in space** (existence of an artifact in different shapes at the same time) Pohl *et al.* claim that variability is important to different stakeholders and thus has different levels of visibility: **external variability** is visible to the customers while **internal variability**, that of domain artifacts, is hidden from them. Other classification proposals come from Meekel *et al.* [129] (feature, hardware platform, performances and attributes variability) or Bass *et al.* [92] who discuss about variability at the architectural level.

Central to the modeling of variability is the notion of *feature*, originally defined by Kang *et al.* as: *a prominent or distinctive user-visible aspect, quality or characteristic of a software system or systems* [125]. Based on this notion of *feature*, they proposed to use a *feature model* to model the variability in a SPL. A feature model consists of a *feature diagram* and other associated information: *constraints* and *dependency rules*. Feature diagrams provide a *graphical tree-like notation depicting the hierarchical organization of high level product functionalities* represented as features. The root of the tree refers to the complete system and is

progressively decomposed into more refined features (tree nodes). Relations between nodes (features) are materialized by *decomposition edges* and *textual constraints*. Variability can be expressed in several ways. Presence or absence of a feature from a product is modeled using *mandatory* or *optional features*. Features are graphically represented as rectangles while some graphical elements (e.g., unfilled circle) are used to describe the variability (e.g., a feature may be optional).

Features can be organized into *feature groups*. Boolean operators *exclusive alternative (XOR)*, *inclusive alternative (OR)* or *inclusive (AND)* are used to select one, several or all the features from a feature group. Dependencies between features can be modeled using *textual constraints*: *requires* (presence of a feature requires the presence of another), *mutex* (presence of a feature automatically excludes another). Feature attributes can be also used for modeling quantitative (e.g., numerical) information. Constraints over attributes and features can be specified as well.

Modeling variability allows an organization to capture and select which version of which variant of any particular aspect is wanted in the system [104]. To implement it cheaply, quickly and safely, redoing by hand the tedious weaving of every aspect is not an option: some form of automation is needed to leverage the modeling of variability [96], [112]. Model Driven Engineering (MDE) makes it possible to automate this weaving process [124]. This requires that models are no longer informal, and that the weaving process is itself described as a program (which is as a matter of facts an executable meta-model [133]) manipulating these models to produce for instance a detailed design that can ultimately be transformed to code, or to test suites [142], or other software artifacts.

3.1.3. Component-based software development

Component-based software development [152] aims at providing reliable software architectures with a low cost of design. Components are now used routinely in many domains of software system designs: distributed systems, user interaction, product lines, embedded systems, etc. With respect to more traditional software artifacts (e.g., object oriented architectures), modern component models have the following distinctive features [111]: description of requirements on services required from the other components; indirect connections between components thanks to ports and connectors constructs [127]; hierarchical definition of components (assemblies of components can define new component types); connectors supporting various communication semantics [107]; quantitative properties on the services [101].

In recent years component-based architectures have evolved from static designs to dynamic, adaptive designs (e.g., SOFA [107], Palladio [97], Frascati [134]). Processes for building a system using a statically designed architecture are made of the following sequential lifecycle stages: requirements, modeling, implementation, packaging, deployment, system launch, system execution, system shutdown and system removal. If for any reason after design time architectural changes are needed after system launch (e.g., because requirements changed, or the implementation platform has evolved, etc) then the design process must be reexecuted from scratch (unless the changes are limited to parameter adjustment in the components deployed).

Dynamic designs allow for *on the fly* redesign of a component based system. A process for dynamic adaptation is able to reapply the design phases while the system is up and running, without stopping it (this is different from stop/redeploy/start). This kind of process supports *chosen adaptation*, when changes are planned and realized to maintain a good fit between the needs that the system must support and the way it supports them [126]. Dynamic component-based designs rely on a component meta-model that supports complex life cycles for components, connectors, service specification, etc. Advanced dynamic designs can also take platform changes into account at run-time, without human intervention, by adapting themselves [109], [155]. Platform changes and more generally environmental changes trigger *imposed adaptation*, when the system can no longer use its design to provide the services it must support. In order to support an eternal system [99], dynamic component based systems must separate architectural design and platform compatibility. This requires support for heterogeneity, since platform evolutions can be partial.

The Models@runtime paradigm denotes a model-driven approach aiming at taming the complexity of dynamic software systems. It basically pushes the idea of reflection one step further by considering the reflection layer as a real model “something simpler, safer or cheaper than reality to avoid the complexity, danger and

irreversibility of reality [146]". In practice, component-based (and/or service-based) platforms offer reflection APIs that make it possible to introspect the system (which components and bindings are currently in place in the system) and dynamic adaptation (by applying CRUD operations on these components and bindings). While some of these platforms offer rollback mechanisms to recover after an erroneous adaptation, the idea of Models@runtime is to prevent the system from actually enacting an erroneous adaptation. In other words, the "model at run-time" is a reflection model that can be uncoupled (for reasoning, validation, simulation purposes) and automatically resynchronized.

Heterogeneity is a key challenge for modern component based system. Until recently, component based techniques were designed to address a specific domain, such as embedded software for command and control, or distributed Web based service oriented architectures. The emergence of the Internet of Things paradigm calls for a unified approach in component based design techniques. By implementing an efficient separation of concern between platform independent architecture management and platform dependent implementations, *Models@runtime* is now established as a key technique to support dynamic component based designs. It provides DIVERSE with an essential foundation to explore an adaptation envelop at run-time.

Search Based Software Engineering [120] has been applied to various software engineering problems in order to support software developers in their daily work. The goal is to automatically explore a set of alternatives and assess their relevance with respect to the considered problem. These techniques have been applied to craft software architecture exhibiting high quality of services properties [117]. Multi Objectives Search based techniques [114] deal with optimization problem containing several (possibly conflicting) dimensions to optimize. These techniques provide DIVERSE with the scientific foundations for reasoning and efficiently exploring an envelope of software configurations at run-time.

3.1.4. Validation and verification

Validation and verification (V&V) theories and techniques provide the means to assess the validity of a software system with respect to a specific correctness envelop. As such, they form an essential element of DIVERSE's scientific background. In particular, we focus on model-based V&V in order to leverage the different models that specify the envelop at different moments of the software development lifecycle.

Model-based testing consists in analyzing a formal model of a system (*e.g.*, activity diagrams, which capture high-level requirements about the system, statecharts, which capture the expected behavior of a software module, or a feature model, which describes all possible variants of the system) in order to generate test cases that will be executed against the system. Model-based testing [154] mainly relies on model analysis, constraint solving [113] and search-based reasoning [128]. DIVERSE leverages in particular the applications of model-based testing in the context of highly-configurable systems and [156] interactive systems [130] as well as recent advances based on diversity for test cases selection [121].

Nowadays, it is possible to simulate various kinds of models. Existing tools range from industrial tools such as Simulink, Rhapsody or Telelogic to academic approaches like Omega [140], or Xholon [2]. All these simulation environments operate on homogeneous environment models. However, to handle diversity in software systems, we also leverage recent advances in heterogeneous simulation. Ptolemy [106] proposes a common abstract syntax, which represents the description of the model structure. These elements can be decorated using different directors that reflect the application of a specific model of computation on the model element. Metropolis [93] provides modeling elements amenable to semantically equivalent mathematical models. Metropolis offers a precise semantics flexible enough to support different models of computation. ModHel'X [119] studies the composition of multi-paradigm models relying on different models of computation.

Model-based testing and simulation are complemented by runtime fault-tolerance through the automatic generation of software variants that can run in parallel, to tackle the open nature of software-intensive systems. The foundations in this case are the seminal work about N-version programming [91], recovery blocks [144] and code randomization [95], which demonstrated the central role of diversity in software to ensure runtime resilience of complex systems. Such techniques rely on truly diverse software solutions in order to provide

[2] http://www.primordion.com/Xholon/

systems with the ability to react to events, which could not be predicted at design time and checked through testing or simulation.

3.1.5. Empirical software engineering

The rigorous, scientific evaluation of DIVERSE's contributions is an essential aspect of our research methodology. In addition to theoretical validation through formal analysis or complexity estimation, we also aim at applying state-of-the-art methodologies and principles of empirical software engineering. This approach encompasses a set of techniques for the sound validation contributions in the field of software engineering, ranging from statistically sound comparisons of techniques and large-scale data analysis to interviews and systematic literature reviews [149], [147]. Such methods have been used for example to understand the impact of new software development paradigms [105]. Experimental design and statistical tests represent another major aspect of empirical software engineering. Addressing large-scale software engineering problems often requires the application of heuristics, and it is important to understand their effects through sound statistical analyses [90].

3.2. Research axis

Figure 1 illustrates the four dimensions of software diversity, which form the core research axis of DIVERSE: the **diversity of languages** used by the stakeholders involved in the construction of these systems; the **diversity of features** required by the different customers; the **diversity of runtime environments** in which software has to run and adapt; the **diversity of implementations** that are necessary for resilience through redundancy. These four axis share and leverage the scientific and technological results developed in the area of model-driven engineering in the last decade. This means that all our research activities are founded on sound abstractions to reason about specific aspects of software systems, compose different perspectives and automatically generate parts of the system.

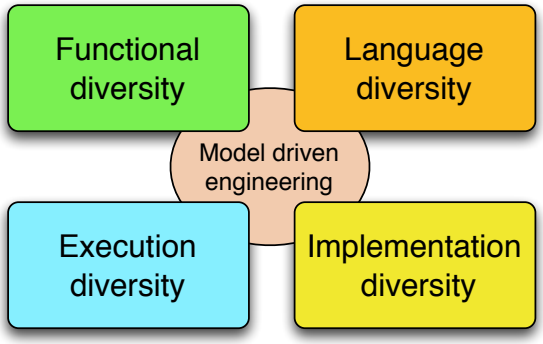

Figure 1. The four research axis of DIVERSE, which rely on a MDE scientific background

3.2.1. Software Language Engineering

The engineering of systems involves many different stakeholders, each with their own domain of expertise. Hence more and more organizations are adopting Domain Specific Modeling Languages (DSMLs) to allow domain experts to express solutions directly in terms of relevant domain concepts [148], [116]. This new trend raises new challenges about designing DSMLs, evolving a set of DSMLs and coordinating the use of multiple DSLs for both DSL designers and DSL users.

3.2.1.1. Challenges

Reusability of software artifacts is a central notion that has been thoroughly studied and used by both academics and industrials since the early days of software construction. Essentially, designing reusable artifacts allows the construction of large systems from smaller parts that have been separately developed and validated, thus reducing the development costs by capitalizing on previous engineering efforts. However, it is still hardly possible for language designers to design typical language artifacts (e.g. language constructs, grammars, editors or compilers) in a reusable way. The current state of the practice usually prevents the reusability of language artifacts from one language to another, consequently hindering the emergence of real engineering techniques around software languages. Conversely, concepts and mechanisms that enable artifacts reusability abound in the software engineering community.

Variability concerns in modeling languages occur in the definition of the abstract and concrete syntax as well as in the specification of the language's semantics. The major challenges met when addressing the need for variability are: (i) set principles for modeling language units that support the modular specification of a modeling language; and (ii) design mechanisms to assemble these units in a complete language, according to the set of authorized variation points for the modeling language family.

A new generation of complex software-intensive systems (for example smart health support, smart grid, building energy management, and intelligent transportation systems) presents new opportunities for leveraging modeling languages. The development of these systems requires expertise in diverse domains. Consequently, different types of stakeholders (e.g., scientists, engineers and end-users) must work in a coordinated manner on various aspects of the system across multiple development phases. DSMLs can be used to support the work of domain experts who focus on a specific system aspect, but they can also provide the means for coordinating work across teams specializing in different aspects and across development phases. The support and integration of DSMLs leads to what we call **the globalization of modeling languages**, *i.e.* the use of multiple languages for the coordinated development of diverse aspects of a system. One can make an analogy with world globalization in which relationships are established between sovereign countries to regulate interactions (e.g., travel and commerce related interactions) while preserving each country's independent existence.

3.2.1.2. Scientific objectives

We address reuse and variability challenges through the investigation of the time-honored concepts of substitutability, inheritance and components, evaluate their relevance for language designers and provide tools and methods for their inclusion in software language engineering. We will develop novel techniques for the modular construction of language extensions with the support of model syntactical variability. From the semantics perspective, we investigate extension mechanisms for the specification of variability in operational semantics, focusing on static introduction and heterogeneous models of computation. The definition of variation points for the three aspects of the language definition provides the foundations for the novel concept Language Unit (LU) as well as suitable mechanisms to compose such units.

We explore the necessary breakthrough in software languages to support modeling and simulation of heterogeneous and open systems. This work relies on the specification of executable domain specific modeling languages (DSMLs) to formalize the various concerns of a software-intensive system, and of models of computation (MoCs) to explicitly model the concurrency, time and communication of such DSMLs. We develop a framework that integrates the necessary foundations and facilities for designing and implementing executable and concurrent domain-specific modeling languages. It also provides unique features to specify composition operators between (possibly heterogeneous) DSMLs. Such specifications are amenable to support the edition, execution, graphical animation and analysis of heterogeneous models. The objective is to provide both a significant improvement of MoCs and DSMLs design and implementation; and the simulation based validation and verification of complex systems.

We see an opportunity for the automatic diversification of programs' computation semantics, for example through the diversification of compilers or virtual machines. The main impact of this artificial diversity is to provide flexible computation and thus ease adaptation to different execution conditions. A combination of static and dynamic analysis could support the identification of what we call *plastic computation zones* in the

code. We identify different categories of such zones: (i) areas in the code in which the order of computation can vary (e.g., the order in which a block of sequential statements is executed); (ii) areas that can be removed, keeping the essential functionality [150] (e.g., skip some loop iterations); (iii) areas that can replaced by alternative code (e.g., replace a try-catch by a return statement). Once we know which zones in the code can be randomized, it is necessary to modify the model of computation to leverage the computation plasticity. This consists in introducing variation points in the interpreter to reflect the diversity of models of computation. Then, the choice of a given variation is performed randomly at run-time.

3.2.2. Variability Modeling and Engineering

The systematic modeling of variability in software systems has emerged as an effective approach to document and reason about software evolutions and heterogeneity (*cf.* Section 3.1.2). Variability modeling characterizes an "envelope" of possible software variations. The industrial use of variability models and their relation to software artifact models require a complete engineering framework, including composition, decomposition, analysis, configuration and artifact derivation, refactoring, re-engineering, extraction, and testing. This framework can be used both to tame imposed diversity and to manage chosen diversity.

3.2.2.1. Challenges

A fundamental problem is that the **number of variants** can be exponential in the number of options (features). Already with 300 boolean configuration options, approximately 10^{90} configurations exist – more than estimated count of atoms in the universe. Domains like automotive or operating systems have to manage more than 10000 options (e.g., Linux). Practitioners face the challenge of developing billions of variants. It is easy to forget a necessary constraint, leading to the synthesis of unsafe variants, or to under-approximate the capabilities of the software platform. Scalable modelling techniques are therefore crucial to specify and reason about a very large set of variants.

Model-driven development supports two ways to deal with the increasing number of concerns in complex systems: (1) multi-view modeling, *i.e.* when modeling each concern separately, and variability modeling. However, there is little support to combine both approaches consistently. Techniques to integrate both approaches will enable the construction of a consistent set of views and variation points in each view.

The design, construction and maintenance of software families have a major impact on **software testing**. Among the existing challenges, we can cite: the selection of test cases for a specific variant; the evolution of test suites with integration of new variants; the combinatorial explosion of the number of software configurations to be tested. Novel model-based techniques for test generation and test management in a software product line context are needed to overcome state-of-the-art limits we already observed in some projects.

3.2.2.2. Scientific objectives

We aim at developing scalable techniques to automatically analyze variability models and their interactions with other views on the software intensive system (requirements, architecture, design). These techniques provide two major advancements in the state of the art: (1) an extension of the semantics of variability models in order to enable the definition of attributes (*e.g.*, cost, quality of service, effort) on features and to include these attributes in the reasoning; (2) an assessment of the consistent specification of variability models with respect to system views (since variability is orthogonal to system modeling, it is currently possible to specify the different models in ways that are semantically meaningless). The former aspect of analysis is tackled through constraint solving and finite-domain constraint programming, while the latter aspect is investigated through automatic search-based techniques (similar to genetic algorithms) for the exploration of the space of interaction between variability and view models.

We aim to develop procedures to reverse engineer dependencies and features' sets from existing software artefacts – be it source code, configuration files, spreadsheets (e.g., product comparison matrices) or requirements. We expect to scale up (e.g., for extracting a very large number of variation points) and guarantee some properties (e.g., soundness of configuration semantics, understandability of ontological semantics). For instance, when building complex software-intensive systems, textual requirements are captured in very large quantities of documents. In this context, adequate models to formalize the organization of requirements documents and automated techniques to support impact analysis (in case of changes in the requirements) have to be developed.

We aim at developing sound methods and tools to integrate variability management in model-based testing activities. In particular, we will leverage requirement models as an essential asset to establish formal relations between variation points and test models. These relations will form the basis for novel algorithms that drive the systematic selection of test configurations that satisfy well-defined test adequacy criteria as well as the generation of test cases for a specific product in the product line.

3.2.3. Heterogeneous and dynamic software architectures

Flexible yet dependable systems have to cope with heterogeneous hardware execution platforms ranging from smart sensors to huge computation infrastructures and data centers. Evolutions range from a mere change in the system configuration to a major architectural redesign, for instance to support addition of new features or a change in the platform architecture (new hardware is made available, a running system switches to low bandwidth wireless communication, a computation node battery is running low, etc). In this context, we need to devise formalisms to reason about the impact of an evolution and about the transition from one configuration to another. It must be noted that this axis focuses on the use of models to drive the evolution from design time to run-time. Models will be used to (i) systematically define predictable configurations and variation points through which the system will evolve; (ii) develop behaviors necessary to handle unpredicted evolutions.

3.2.3.1. Challenges

The main challenge is to provide new homogeneous architectural modelling languages and efficient techniques that enable continuous software reconfiguration to react to changes. This work handles the challenges of handling the diversity of runtime infrastructures and managing the cooperation between different stakeholders. More specifically, the research developed in this axis targets the following dimensions of software diversity.

Platform architectural heterogeneity induces a first dimension of imposed diversity (type diversity). Platform reconfigurations driven by changing resources define another dimension of diversity (deployment diversity). To deal with these imposed diversity problems, we will rely on model based runtime support for adaptation, in the spirit of the dynamic distributed component framework developed by the Triskell team. Since the runtime environment composed of distributed, resource constrained hardware nodes cannot afford the overhead of traditional runtime adaptation techniques, we investigate the design of novel solutions relying on models@runtime and on specialized tiny virtual machines to offer resource provisioning and dynamic reconfigurations. In the next two years this research will be supported by the InfraJVM project.

Diversity can also be an asset to optimize software architecture. Architecture models must integrate multiple concerns in order to properly manage the deployment of software components over a physical platform. However, these concerns can contradict each other (*e.g.*, accuracy and energy). In this context, we investigate automatic solutions to explore the set of possible architecture models and to establish valid trade-offs between all concerns in case of changes.

3.2.3.2. Scientific objectives

Automatic synthesis of optimal software architectures. Implementing a service over a distributed platform (*e.g.*, a pervasive system or a cloud platform) consists in deploying multiple software components over distributed computation nodes. We aim at designing search-based solutions to (i) assist the software architect in establishing a good initial architecture (that balances between different factors such as cost of the nodes, latency, fault tolerance) and to automatically update the architecture when the environment or the system itself change. The choice of search-based techniques is motivated by the very large number of possible software deployment architectures that can be investigated and that all provide different trade-offs between qualitative factors. Another essential aspect that is supported by multi-objective search is to explore different architectural solutions that are not necessarily comparable. This is important when the qualitative factors are orthogonal to each other, such as security and usability for example.

Flexible software architecture for testing and data management. As the number of platforms on which software runs increases and different software versions coexist, the demand for testing environments also increases. For example, to test a software patch or upgrade, the number of testing environments is the product of the number of running environments the software supports and the number of coexisting versions of the

software. Based on our first experiment on the synthesis of cloud environment using architectural models, our objective is to define a set of domain specific languages to catch the requirement and to design cloud environments for testing and data management of future internet systems from data centers to things. These languages will be interpreted to support dynamic synthesis and reconfiguration of a testing environment.

Runtime support for heterogeneous environments. Execution environments must provide a way to account or reserve resources for applications. However, current execution environments such as the Java Virtual Machine do not clearly define a notion of application: each framework has its own definition. For example, in OSGi, an application is a component, in JEE, an application is most of the time associated to a class loader, in the Multi-Tasking Virtual machine, an application is a process. The challenge consists in defining an execution environment that provides direct control over resources (CPU, Memory, Network I/O) independently from the definition of an application. We propose to define abstract resource containers to account and reserve resources on a distributed network of heterogeneous devices.

3.2.4. Diverse implementations for resilience

Open software-intensive systems have to evolve over their lifetime in response to changes in their environment. Yet, most verification techniques assume a closed environment or the ability to predict all changes. Dynamic changes and evolutions thus represent a major challenge for these techniques that aim at assessing the correctness and robustness of the system. On the one hand, DIVERSE will adapt V&V techniques to handle diversity imposed by the requirements and the execution environment, on the other hand we leverage diversity to increase the robustness of software in face of unpredicted situations. More specifically, we address the following V&V challenges.

3.2.4.1. Challenges

One major challenge to build flexible and open yet dependable systems is that current software engineering techniques require architects to foresee all possible situations the system will have to face. However, openness and flexibility also mean unpredictability: unpredictable bugs, attacks, environmental evolutions, etc. Current fault-tolerance [144] and security [115] techniques provide software systems with the capacity of detecting accidental and deliberate faults. However, existing solutions assume that the set of bugs or vulnerabilities in a system does not evolve. This assumption does not hold for open systems, thus it is essential to revisit fault-tolerance and security solutions to account for diverse and unpredictable faults.

Diversity is known to be a major asset for the robustness of large, open, and complex systems (*e.g.*, economical or ecological systems). Following this observation, the software engineering literature provides a rich set of work that choose to implement diversity in software systems in order to improve robustness to attacks or to changes in quality of service. These works range from N-version programming to obfuscation of data structures or control flow, to randomization of instruction sets. An essential remaining challenge is to support the automatic synthesis and evolution of software diversity in open software-intensive systems. There is an opportunity to further enhance these techniques in order to cope with a wider diversity of faults, by multiplying the levels of diversity in the different software layers that are found in software-intensive systems (system, libraries, frameworks, application). This increased diversity must be based on artificial program transformations and code synthesis, which increase the chances of exploring novel solutions, better fitted at one point in time. The biological analogy also indicates that diversity should emerge as a side-effect of evolution, to prevent over-specialization towards one kind of diversity.

3.2.4.2. Scientific objectives

The main objective is to address one of the main limitations of N-version programming for fault-tolerant systems: the manual production and management of software diversity. Through automated injection of artificial diversity we aim at systematically increasing failure diversity and thus increasing the chances of early error detection at run-time. A fundamental assumption for this work is that software-intensive systems can be “good enough” [145], [157].

Proactive program diversification. We aim at establishing novel principles and techniques that favor the emergence of multiple forms of software diversity in software-intensive systems, in conjunction with the software adaptation mechanisms that leverage this diversity. The main expected outcome is a set of meta-design principles that maintain diversity in systems and the experimental demonstration of the effects of software diversity on the adaptive capacities of CASs. Higher levels of diversity in the system provide a pool of software solutions that can eventually be used to adapt to situations unforeseen at design time (bugs, crash, attacks, etc.). Principles of automated software diversification rely on the automated synthesis of variants in a software product line, as well as finer-grained program synthesis combining unsound transformations and genetic programming to explore the space of mutational robustness.

Multi-tier software diversification. We call multi-tier diversification the fact of diversifying several application software components simultaneously. The novelty of our proposal, with respect to the software diversity state of the art, is to diversify the application-level code (for example, diversify the business logics of the application), focusing on the technical layers found in web applications. The diversification of application software code is expected to provide a diversity of failures and vulnerabilities in web server deployment. Web server deployment usually adopts a form of the Reactor architecture pattern, for scalability purposes: multiple copies of the server software stack, called request handlers, are deployed behind a load balancer. This architecture is very favorable for diversification, since by using the multiplicity of request handlers running in a web server we can simultaneously deploy multiple combinations of diverse software components. Then, if one handler is hacked or crashes the others should still be able to process client requests.

4. Application Domains

4.1. From Embedded Systems to Service Oriented Architectures

From small embedded systems such as home automation products or automotive systems to medium sized systems such as medical equipment, office equipment, household appliances, smart phones; up to large Service Oriented Architectures (SOA), building a new application from scratch is no longer possible. Such applications reside in (group of) machines that are expected to run continuously for years without unrecoverable errors. Special care has then to be taken to design and validate embedded software, making the appropriate trade-off between various extra-functional properties such as reliability, timeliness, safety and security but also development and production cost, including resource usage of processor, memory, bandwidth, power, etc.

Leveraging ongoing advances in hardware, embedded software is playing an evermore crucial role in our society, bound to increase even more when embedded systems get interconnected to deliver ubiquitous SOA. For this reason, embedded software has been growing in size and complexity at an exponential rate for the past 20 years, pleading for a component based approach to embedded software development. There is a real need for flexible solutions allowing to deal at the same time with a wide range of needs (product lines modeling and methodologies for managing them), while preserving quality and reducing the time to market (such as derivation and validation tools).

We believe that building flexible, reliable and efficient embedded software will be achieved by reducing the gap between executable programs, their models, and the platform on which they execute, and by developing new composition mechanisms as well as transformation techniques with a sound formal basis for mapping between the different levels.

Reliability is an essential requirement in a context where a huge number of softwares (and sometimes several versions of the same program) may coexist in a large system. On one hand, software should be able to evolve very fast, as new features or services are frequently added to existing ones, but on the other hand, the occurrence of a fault in a system can be very costly, and time consuming. While we think that formal methods may help solving this kind of problems, we develop approaches where they are kept “behind the scene” in a global process taking into account constraints and objectives coming from user requirements.

Software testing is another aspect of reliable development. Testing activities mostly consist in trying to exhibit cases where a system implementation does not conform to its specifications. Whatever the efforts spent for development, this phase is of real importance to raise the confidence level in the fact that a system behaves properly in a complex environment. We also put a particular emphasis on on-line approaches, in which test and observation are dynamically computed during execution.

5. New Software and Platforms

5.1. Kermeta

Participants: Benoit Combemale [correspondant], Olivier Barais, Arnaud Blouin, Didier Vojtisek, Benoit Baudry, Thomas Degueule, David Mendez, Erwan Bousse, Francois Tanguy, Fabien Coulon.

Nowadays, object-oriented meta-languages such as MOF (Meta-Object Facility) are increasingly used to specify domain-specific languages in the model-driven engineering community. However, these meta-languages focus on structural specifications and have no built-in support for specifications of operational semantics. Integrated with the Ecore industrial standard and aligned with the EMOF 2.0 OMG standard, the Kermeta language consists in a extension to these meta-languages to support behavior definition. The language adds precise action specifications with static type checking and genericity at the meta level. Based on object-orientation and aspect orientation concepts, the Kermeta language adds model specific concepts.

Kermeta is used in several use cases:

- to give a precise semantic of the behavior of a metamodel, which then can be simulated;
- to act as a model transformation language;
- to act as a constraint language.

In 2014, we have continued the refactoring of Kermeta to leverage on Xtend. The Kermeta action language is now defined as an extension of Xtend, by proposing model-specific features (e.g., model type, containment, opposite) and an open class mechanism for aspect weaving. The main objective of this new refactoring was to benefit from the non model–specific features of Xtend (including the basics of the action language and its respective tooling such as editor, type checker and compiler), and to focus in our development on innovative solutions for MDE.

More precisely, in addition to an Xtend extension dedicated to model manipulation, we started to integrate in Kermeta various facilities to support software language engineering (slicing, pruning, reuse, variability management, etc). In 2014, we improved this software language engineering feature (currently named k3sle/Melange) in order to offer a functional model typing system allowing safe model polymorphism. This system enables reuse of algorithms and transformations accross different metamodels, as well as language inheritance, evolution and interoperability.

Moreover, while this version of Kermeta is a DSML development workbench that provides good support for developping independent DSMLs, little or no support is provided for integrated use of multiple DSMLs. The lack of support for explicitly relating concepts expressed in different DSMLs makes it very difficult for developers to reason about information spread across models describing different system aspects.

According to Google Scholar [3], the Kermeta platform was used or cited in more than 1300 papers.

5.2. FAMILIAR

Participants: Mathieu Acher [correspondant], Olivier Barais, Guillaume Bécan, Aymeric Hervieu, Julien Richard-Foy, Sana Ben Nasr, Edward Mauricio Alferez Salinas, João Ferreira Filho, Didier Vojtisek, Benoit Baudry.

[3] http://scholar.google.fr/scholar?q=kermeta+model

Modeling and reasoning about configuration options is crucial for the effective management of configurable software systems and product lines. The FAMILIAR project provides dedicated languages, APIs, and comprehensive environments for that purpose. Specifically, FAMILIAR provides support for feature models (by far the most popular notation). The feature models formalism has been studied for more than 20 years [98], and it is widely used in the industry [100]. FAMILIAR (for FeAture Model scrIpt Language for manIpulation and Automatic Reasoning) provides a scripting language for importing, exporting, composing, decomposing, editing, configuring, computing "diffs", refactoring, reverse engineering, testing, and reasoning about (multiple) feature models. For interoperability, many bridges with existing feature modeling languages are implemented. All these operations can be combined to perform complex variability management tasks: extraction of feature models from software artifacts [87], product line evolution [89], management of multiple models [88] [75], [76], model-based validation of SPLs [22], large scale configuration of feature models [122], etc. The level of maturity of the FAMILIAR platform is TRL 3 (*i.e.* New technology tested Prototype built and functionality demonstrated through testing over a limited range of operating conditions. These tests can be done on a scaled version if scalable).

Main innovative features:

- reverse engineering of variability models from multiple kinds of artefacts;
- composition of multiple variability models (e.g., for combining different sources of variability);
- slicing of variability model (e.g., for scheduling a configuration process in different steps);
- connection with the Common Variability Language (CVL);
- support of advanced variability constructs (e.g., attributes, multi-features, meta-information);
- Web-based, comprehensive environment (WebFML [42]).

Impact:

The results are connected to the CVL standardization initiative. From a research perspective, FAMILIAR helps to support all the research activity on variability modeling (e.g., design of new operators, benchmarking). Several tutorials and tool demonstrations [42], [25] have been performed at SPLC (the major conference in software product lines), at ECOOP, at CIEL and MODELS in 2012 and 2013. FAMILIAR is also used in the context of teaching activities. From an industrial perspective, the languages and tools have already been applied in practical contexts in different application domains (medical imaging, video surveillance, system engineering, web configurators, etc.) and for various purposes. This platform is also used for supporting industrial transfer activity with companies such as Thales. FAMILIAR is involved in several research projects (e.g., in the Merge ITEA project, in the MOTIV project, in the VaryMDE project).

FAMILIAR is distributed under the terms of the LGPL and EPL open source license.

See also the web page familiar-project.github.com.

- Version: 1.3
- Programming language: Java, Scala

5.3. Kevoree

Participants: Olivier Barais [correspondant], Johan Bourcier, Noel Plouzeau, Benoit Baudry, Maxime Tricoire, Jacky Bourgeois, Inti G. Herrera, Ivan Paez, Francisco Acosta, Mohamed Boussaa.

Kevoree is an open-source models@runtime platform [4] to properly support the dynamic adaptation of distributed systems. Models@runtime basically pushes the idea of reflection [132] one step further by considering the reflection layer as a real model that can be uncoupled from the running architecture (e.g. for reasoning, validation, and simulation purposes) and later automatically resynchronized with its running instance.

[4] http://www.kevoree.org

Kevoree has been influenced by previous work that we carried out in the DiVA project [132] and the Entimid project [135]. With Kevoree we push our vision of models@runtime [131] farther. In particular, Kevoree provides a proper support for distributed models@runtime. To this aim we introduced the *Node* concept to model the infrastructure topology and the *Group* concept to model semantics of inter node communication during synchronization of the reflection model among nodes. Kevoree includes a *Channel* concept to allow for multiple communication semantics between remote*Components* deployed on heterogeneous nodes. All Kevoree concepts (Component, Channel, Node, Group) obey the object type design pattern to separate deployment artifacts from running artifacts. Kevoree supports multiple kinds of very different execution node technology (e.g. Java, Android, MiniCloud, FreeBSD, Arduino, ...).

Kevoree is distributed under the terms of the LGPL open source license.

Main competitors:

- the Fractal/Frascati eco-system [5].
- SpringSource Dynamic Module [6]
- GCM-Proactive [7]
- OSGi [8]
- Chef [9]
- Vagran [10]

Main innovative features:

- distributed models@runtime platform (with a distributed reflection model and an extensible models@runtime dissemination set of strategies).
- Support for heterogeneous node type (from Cyber Physical System with few resources until cloud computing infrastructure).
- Fully automated provisioning model to correctly deploy software modules and their dependencies.
- Communication and concurrency access between software modules expressed at the model level (not in the module implementation).

Impact:

A tutorial have been performed at the Comparch conference in July 2014 and at the Middleware conference in december 2014. See also the web page http://www.kevoree.org.

In 2014, we mainly created a new implementation in JavaScript and we created an implementation for system containers for driving resources using Kevoree. We also use Kevoree in the context of Mohammed's PhD to create testing infrastructure on-demand.

- Version: 5.1.4
- Programming language: Java, Scala, Kermeta, Kotlin, Javascript

6. New Results

6.1. Highlights of the Year

"Globalizing Modeling Languages" appears in IEEE Computer Magazine. This paper synthesizes our vision of how domain-specific languages form the foundations of global software development. Its appearance in a highly visible venue is major milestone for the dissemination and impact of our work about the diversity of languages.

[5] http://frascati.ow2.org
[6] http://spring.io/
[7] http://proactive.inria.fr/
[8] http://www.osgi.org
[9] http://wiki.opscode.com/display/chef/Deploy+Resource
[10] http://vagrantup.com/

DiverSE extremely present at the SPLC conference. SPLC is the main international conference for software product line engineering. In 2014, the DiverSE team had a very strong presence at this conference, presenting novel scientific contributions, results of industrial collaborations, and demonstrations of latest software tools.

6.2. Results on Software Language Engineering

The engineering of systems involves many different stakeholders, each with their own domain of expertise. Hence more and more organizations are adopting Domain Specific Languages (DSLs) to allow domain experts to express solutions directly in terms of relevant domain concepts. This new trend raises new challenges about designing DSLs, evolving a set of DSLs and coordinating the use of multiple DSLs for both DSL designers and DSL users. In [56] we present the overall vision that we develop in the DiverSE team about Software Language Engineering. The main results on this topic are presented below.

6.2.1. Globalization of Domain Specific Languages

In the software and systems modeling community, research on domain-specific modeling languages (DSMLs) focuses on technologies for developing languages and tools to increase the effectiveness of domain experts. Yet, there is a lack of support to explicitly relate concepts expressed in different DSMLs, which prevents software and system engineers to reason about information spread across models describing different system aspects. Supporting coordinated use of DSMLs leads to what we call the globalization of modeling languages [20]. In such a context, we develop a research initiative that broadens the DSML research focus beyond the development of independent DSMLs to one that supports globalized DSMLs, that is, DSMLs that facilitate coordination of work across different domains of expertise. We also provid a formal framework to prove the correctness of model driven engineering composition operators [57].

6.2.2. Meta-Language for the Concurrency Concern in DSLs

Concurrency is of primary interest in the development of complex software-intensive systems, as well as the deployment on modern platforms. However, reifying the definition of the DSL concurrency remains a challenge. This hinders: a) the development of a complete understanding of the DSL semantics; b) the effectiveness of concurrency-aware analysis techniques; c) the analysis of the deployment on parallel architectures. In this context, we present MoCCML, a dedicated meta-language for formally specifying the concurrency concern within the definition of a DSL [44]. The concurrency constraints can reflect the knowledge in a particular domain, but also the constraints of a particular platform. MoCCML comes with a complete language workbench to help a DSL designer in the definition of the concurrency directly within the concepts of theDSL itself, and a generic workbench to simulate and analyze any model conforming to this DSL. Mo CCML is illustrated on the definition of an lightweight extension of SDF (SynchronousData Flow).

6.2.3. Automating Variability Model Inference for Component-Based Language Implementations

Componentized language frameworks, coupled with variability modeling, have the potential to bring language development to the masses, by simplifying the configuration of a new language from an existing set of reusable components. However, designing variability models for this purpose requires not only a good understanding of these frameworks and the way components interact, but also an adequate familiarity with the problem domain. In [68] we propose an approach to automatically infer a relevant variability model from a collection of already implemented language components, given a structured, but general representation of the domain. We describe techniques to assist users in achieving a better understanding of the relationships between language components, and find out which languages can be derived from them with respect to the given domain.

6.2.4. Metamorphic Domain-Specific Languages

External or internal domain-specific languages (DSLs) or (fluent) APIs? Whoever you are – a developer or a user of a DSL – you usually have to choose side; you should not! What about metamorphic DSLs that change their shape according to your needs? Our 4-years journey of providing the "right" support (in the domain of feature modeling), led us to develop an external DSL, different shapes of an internal API, and maintain

all these languages. A key insight is that there is no one-size-fits-all solution or no clear superiority of a solution compared to another. On the contrary, we found that it does make sense to continue the maintenance of an external and internal DSL. Based on our experience and on an analysis of the DSL engineering field, the vision that we foresee for the future of software languages is their ability to be self-adaptable to the most appropriate shape (including the corresponding integrated development environment) according to a particular usage or task. We call metamorphic DSL such a language, able to change from one shape to another shape [27].

6.2.5. Adapting mutation testing for model transformations

Due to the specificities of models and transformations, classical software testing techniques have to be adapted. Among these techniques, mutation analysis has been ported and a set of mutation operators has been defined. However, mutation analysis currently requires a considerable manual work and is hampered by the test data set improvement activity. This activity is seen by testers as a difficult and time-consuming job, and reduces the benefits of the mutation analysis.

We provide a model transformation traceability mechanism, in conjunction with a model of mutation operators and a dedicated algorithm, to automatically or semi-automatically produce test models that detect new faults [18].

6.2.6. Efficient model cloning for analysis

We propose an original approach that exploits the fact that operations rarely modify a whole model. Given a set of immutable properties, our cloning approach determines the objects and fields that can be shared between the runtime representations of a model and its clones. Our generic cloning algorithm is parameterized with three strategies that establish a trade-off between memory savings and the ease of clone manipulation. We evaluated memory footprints and computation overheads with 100 randomly generated metamodels and models [40]. We have also drawn the research roadmap to exploit these efficient clone operations to analyze multidimensional execution traces [41].

6.3. Results on Variability Modeling and Engineering

6.3.1. Engineering Interactive Systems

In agreement with our permanent effort to validate the techniques we propose on real use cases in various domains, we applied seminal MDE to interactive systems engineering. This led to two collaborations. The first one has been conducted with 3D Collaborative Virtual Environments (3D CVE) researchers. Despite the increasing use of 3D CVE, their development is still a cumbersome task. The various concerns to consider (distributed system, 3D graphics, *etc.*) complexify their development as well as their evolution. We propose to leverage MDE for developing 3D CVEs [45]. We have shown how a 3D CVE framework benefits from a DSL we built using state-of-the-art MDE technologies. The benefits are multiple: 3D CVEs designers can focus on the behavior of their virtual objects without bothering with distributed and graphics features; configuring the content of 3D CVEs and their deployment on various software and hardware platforms can be automated through code generation.

The second collaboration is international and has been conducted with software visualization researchers. Current metamodel editing tools are based on standard visualization and navigation features, such as physical zooms. However, as soon as metamodels become larger, navigating through large metamodels becomes a tedious task that hinders their understanding. In this work, we promote the use of model slicing techniques [102] to build visualization techniques dedicated to metamodels [37]. This approach is implemented in a metamodel visualizer, called `Explain`.

6.3.2. Variability management in regulatory requirements and system engineering

Nuclear power plants are some of the most sophisticated and complex energy systems ever designed. These systems perform safety critical functions and must conform to national safety institutions and international regulations. In many cases, regulatory documents provide very high level and ambiguous requirements that leave a wide margin for interpretation. As the French nuclear industry is now seeking to spread its activities outside France, it is but necessary to master the ins and the outs of the variability between countries safety culture and regulations. This sets both an industrial and a scientific challenge to introduce and propose a product line engineering approach to an unaware industry whose safety culture is made of interpretations, specificities, and exceptions. We have developed two contributions within the French R&D project CONNEXION, while introducing variability modeling to the French nuclear industry [66], [34].

As part of the VaryMDE project (a bilateral collaboration between Thales and Inria) we have developed techniques to generate counter-examples (also called anti-patterns) of model-based product lines [22]. The goal is to infer (1) guidelines or domain-specific rules to avoid earlier the specification of incorrect mappings (2) testing oracles for increasing the robustness of derivation engines given a modeling language. We have applied the approach in the context of a real industrial scenario with Thales involving a large-scale metamodel.

6.3.3. Handling testing challenges in product line engineering

Testing techniques in industry are not yet adapted for product line engineering (PLE).

We have developed original contributions to adapt model-based testing for PLE [65], [63], [13]. We equip usage models, a widely used formalism in MBT, with variability capabilities. Formal correspondences are established between a variability model, a set of functional requirements, and a usage model. An algorithm then exploits the traceability links to automatically derive a usage model variant from a desired set of selected features. The approach is integrated into the MBT tool MaTeLo and is currently used in industry.

We have also developed a variability-based testing approach to derive video sequence variants. The ideas of our VANE approach are i) to encode in a variability model what can vary within a video sequence; ii) to exploit the variability model to generate testable configurations; iii) to synthesize variants of video sequences corresponding to configurations. VANE computes T-wise covering sets while optimizing a function over attributes [50], [25].

6.3.4. Reverse engineering variability models

We have developed automated techniques and a comprehensive environment for synthesizing feature models from various kinds of artefacts (e.g. propositional formula, dependency graph, FMs or product comparison matrices). Specifically we have elaborated a support (through ranking lists, clusters, and logical heuristics) for choosing a sound and meaningful hierarchy [42]. We have performed an empirical evaluation on hundreds of feature models, coming from the SPLOT repository and Wikipedia [108]. We have showed that a hybrid approach mixing logical and ontological techniques outperforms state-of-the-art solutions (to appear in Empirical Software Engineering journal in 2015 [19]). Beyond the reverse engineering of variability, our work has numerous practical applications (e.g., merging multiple product lines, slicing a configuration process).

6.3.5. Product comparison matrices

Product Comparison Matrices (PCMs) constitute a rich source of data for comparing a set of related and competing products over numerous features. Despite their apparent simplicity, PCMs contain heterogeneous, ambiguous, uncontrolled and partial information that hinders their efficient exploitations. We have first elaborated our vision and identify research challenges for an exploitation of PCMs when engineering comparators, configurators, or other services [67].

We have formalized PCMs through model-based automated techniques and developed additional tooling to support the edition and re-engineering of PCMs [43]. 20 participants used our editor to evaluate our PCM metamodel and automated transformations. The empirical results over 75 PCMs from Wikipedia show that (1) a significant proportion of the formalization of PCMs can be automated: 93.11% of the 30061 cells are

correctly formalized; (2) the rest of the formalization can be realized by using the editor and mapping cells to existing concepts of the metamodel.

The ASE'2014 paper opens avenues for engaging a community in the mining, re-engineering, edition, and exploitation of PCMs that now abound on the Internet. We have launched an open, collaborative initiative towards this direction http://www.opencompare.org

6.4. Results on Heterogeneous and dynamic software architectures

This year, we focused on the challenges that use *models@runtime* for resource-constrained and resource-aware systems. Our main results are in the following four subdomains:

- We designed an adaptive monitoring framework for component-based systems in which we highlight the benefits of using models@runtime for adaptive monitoring.
- We improved models@runtime technologies for resource-constrained devices.
- We designed efficient reasoning techniques for dynamic software architecture, focusing in particular on resource consumption optimization challenges.
- We performed several experiments on the Internet of Things application domain.

The next section details our experiments.

6.4.1. Resource-aware dynamic architecture

Modern component frameworks support continuous deployment and simultaneous execution of multiple software components on top of the same virtual machine. However, isolation between the various components is limited. A faulty version of any one of the software components can compromise the whole system by consuming all available resources. We propose a solution to efficiently identify faulty software components running simultaneously in a single virtual machine. It is based on an optimistic adaptive monitoring system to identify the faulty component. Suspected components are instrumented to obtain fined grain data for deeper analysis by the monitoring system, but only when required. Unsuspected components are left untouched and execute normally. Thus, we perform localized, just-in-time monitoring that decreases the accumulated overhead of the monitoring system. We evaluated our approach against a state-of-the-art monitoring system and we have shown that our technique correctly detects faulty components, while reducing overhead by an average of 80% [52]. Based on this work, we have presented two tutorials at the CBSE/QoSA conference [49] and at the Middleware conference [51].

6.4.2. Technology enablers for resource-aware dynamic software architecture

Models@runtime provides semantically rich reflection layers enabling intelligent systems to reason about themselves and their surrounding context. Most reasoning processes require not only to explore the current state, but also the past history to take sustainable decisions e.g. to avoid oscillating between states. Models@runtime and model-driven engineering in general lack native mechanisms to efficiently support the notion of history, and current approaches usually generate redundant data when versioning models, which reasoners need to navigate. Because of this limitation, models fail in providing suitable and sustainable abstractions to deal with domains relying on history-aware reasoning. This work tackles this challenge by considering history as a native concept for modeling foundations. Integrated in conjunction with lazy load/storage techniques into the Kevoree Modeling Framework, we demonstrated onto a energy-aware smart grid case study that this mechanisms enable a sustainable reasoning about massive historized models [53].

In this field we also created a specific extension to the `docker.io` open-source project to support a dynamic resource reservation of running containers [9]

6.4.3. Efficient reasoning techniques for dynamic software architecture

Providing software with the capacity of adapting itself according to its environment requires effective techniques to reason and decide on what adaptation to undertake over the running system. To decide on a system adaptation, we have to characterize the value of the system in its corresponding execution environment. A system cannot be characterized by a single dimension, but only using several dimensions such as performance, energy consumption, security and so on. In this context, we have proposed various techniques to leverage multi-objective evolutionary algorithms both at deployment time [46], [21] and at runtime [47] to enable system optimization using multidimensional optimization. We have also proposed a technique to adapt a system proactively based on predictions in order to prevent failures [60]

6.4.4. The Internet of Things application domain

We apply our techniques for heterogeneous and dynamic software architecture more specifically to the Internet of Things application domain. We have two main contributions: (1) an application of the models@runtime concepts on embedded nodes with very limited resources for memory, CPU and battery [30], and (2) a study on the problem of renewable energy production and consumption at home [39]. Domestic microgeneration is the onsite generation of low and zero-carbon heat and electricity by private households to meet their own needs. In this paper we explore how an everyday household routine (doing laundry) can be augmented by digital technologies to help households with photovoltaic solar energy generation to make better use of self-generated energy. We present an 8 month in the field study that involved 18 UK households in longitudinal energy data collection, prototype deployment and participatory data analysis [38]. Through a series of technology interventions mixing energy feedback, proactive suggestions and direct control, the study uncovered opportunities, potential rewards and barriers for families to shift energy consuming household activities. The study highlights how digital technology can act as a mediator between household laundry routines and energy demand-shifting behaviors. Finally, the study provides insights into how a "smart" energy-aware washing machine shapes organization of domestic life and how people "communicate" with their washing machine.

6.5. Results on Diverse Implementations for Resilience

Diversity is acknowledged as a crucial element for resilience, sustainability and increased wealth in many domains such as sociology, economy and ecology. Yet, despite the large body of theoretical and experimental science that emphasizes the need to conserve high levels of diversity in complex systems, the limited amount of diversity in software-intensive systems is a major issue. This is particularly critical as these systems integrate multiple concerns, are connected to the physical world through multiple sensors, run eternally and are open to other services and to users. Here we present our latest observational and technical results about new approaches to increase diversity in software systems.

6.5.1. Automatic synthesis of computationally diverse program variants

The predictability of program execution provides attackers with a rich source of knowledge that they can exploit to spy or remotely control the program. Moving target defense addresses this issue by constantly switching between many diverse variants of a program, thus reducing the certainty that an attacker can have about the program execution. The effectiveness of this approach relies on the availability of a large number of software variants that exhibit different executions. However, current approaches rely on the natural diversity provided by off-the-shelf components, which is very limited. We have explored the automatic synthesis of large sets of program variants, called *sosies* [32]. Sosies provide the same expected functionality as the original program, while exhibiting different executions. They are said to be computationally diverse.

6.5.2. Software Evolution for Diversity Emergence

We aim at favoring spontaneous diversification in software systems, to increase their adaptive capacities. This objective is founded on three observations: (1) software has to constantly evolve to face unpredictable changes in its requirements, execution environment or to respond to failure (bugs, attacks, etc.); (2) the emergence and maintenance of high levels of diversity are essential to provide adaptive capacities to many forms of complex

systems, ranging from ecological and biological systems to social and economical systems; (3) diversity levels tend to be very low in software systems. In this work [33], we consider evolution as a driver for diversity as a means to increase resilience in software systems. In particular, we are inspired by bipartite ecological relationships to investigate the automatic diversification of the server side of a client-server architecture.

6.5.3. Analyzing the diversity of development practices in open source projects

Decentralized version control systems allow a rich structure of commit histories, which presents features that are typical of complex graph models. We bring some evidences of how the very structure of these commit histories carries relevant information about the distributed development process. By means of a novel data structure that we formally define, we analyze the topological characteristics of commit graphs of a sample of git projects. Our findings point out the existence of common recurrent structural patterns that identically occur in different projects and can be considered building blocks of distributed collaborative development [36], [35].

7. Bilateral Contracts and Grants with Industry

7.1. Bilateral Contracts with Industry

7.1.1. Bilateral industrial partnerships

7.1.1.1. Partnership with Thales

Dates: 2011-2014

This partnership with Thales Research and Technology explores variability management both in modeling and metamodeling (*i.e.*, design and implementation of software languages). At the model level, this collaboration is a direct follow-up of the MOVIDA and the MUTATION projects, in which we explore the challenges related to software product line and multi-view engineering for the different development phases of systems of systems construction. At the metamodeling level, we investigate how the notions of variability modeling and management can serve the rigorous definition of families of modeling languages, which address the different interpretations of UML needed to model the different viewpoints in the systems engineering.

The project enrolls 4 faculty members and 2 PhD students from the Triskell team. This year, we keep working on the CVL usage in the Thales context.

7.1.1.2. CIFRE grants

- All4Tec (2011-2014). In this project with the All4Tec company we investigate the support of variability modelling for model-based test generation with Matelo (a tool developed by All4Tec). In this context, Benoit Baudry acts as Ph.D advisor for Hamza Samih.
- Zenexity (2011-2014). In this project with the Zenexity company we investigate the new architecture model for efficient web development on top of the play framework (a web framework developed by Zenexity). In this context, Jean-Marc Jézéquel and Olivier Barais act as Ph.D advisor for Julien Richard Foy.
- Orange (2014-2017). In this project with the Orange company we investigate the support of trust in web communication using software reconfiguration techniques. In this context, Olivier Barais acts as Ph.D advisor for Kevin Corre with Gerson Sunye.

8. Partnerships and Cooperations

8.1. Regional Initiatives

We obtained a grant from the Britany region, within the plan of action SAD (for "Stratégie d'attractivité durable"). The VIP project (for "Visualisation Interactive de produits dans les configurateurs") aims to investigate software product line techniques. We have recruited Dr. Jin Kin (post-doc for a duration of 18 months, starting in december 2014) in collaboration with the ESTASYS team.

8.2. National Initiatives

8.2.1. ANR

8.2.1.1. ANR GEMOC

- Coordinator: Inria (DIVERSE)
- Other partners: ENSTA Bretagne, Inria, IRIT, I3S, Obeo, Thales
- Dates: 2012-2016
- Abstract: GEMOC focuses on a generic framework for heterogeneous software model execution and dynamic analysis. This work has the ambition to propose an innovative environment for the design of complex software-intensive systems by providing: a formal framework that integrates state-of-the-art in model-driven engineering (MDE) to build domain-specific modeling languages (DSMLs), and models of computation (MoC) to reason over the composition of heterogeneous concerns; an open-source design and modeling environment associated to a well-defined method for the definition of DSMLs, MoCs and rigorous composition of all concerns for execution and analysis purposes.

  This requires addressing two major scientific issues: the design and verification of a formal framework to combine several different DSMLs relying on distinct MoCs; the design and validation of a methodology for DSMLs and MoC development. GEMOC aims at participating in the development of next generation MDE environments through a rigorous, tool-supported process for the definition of executable DSMLs and the simulation of heterogeneous models.

8.2.1.2. ANR INFRA-JVM

- Coordinator: Université Paris 6
- Other partners: Université Bordeaux 1, Université Rennes 1 (DIVERSE), Ecole des Mines de Nantes
- Dates: 2012-2015
- Abstract: INFRA-JVM is an ANR project whose goal is to design and provide a new Java Virtual Machine dedicated to pervasive environments. This project focuses on designing a Java Virtual Machine for embedded computing platform offering dynamic reconfiguration capabilities. In this context, DIVERSEaddresses the problem of efficiently identifying faulty software components running simultaneously in a virtual machine without isolation. Current solutions that perform permanent and extensive monitoring to detect anomalies induce very high overhead on the system, and can, by themselves, make the system unstable. Our main objective is to investigate an optimistic adaptive monitoring system using models@runtime to determine the faulty components of an application.

8.2.1.3. SOPRANO

- Coordinator: CEA
- CEA, University of Paris-Sud, Inria Rennes, OcamlPro, Adacore
- Dates: 2014-2017
- Abstract: Today most major verification approaches rely on automatic external solvers. However these solvers do not fill the current and future needs for verification: lack of satisfying model generation, lack of reasoning on difficult theories (e.g. floating-point arithmetic), lack of extensibility for specific or new needs. The SOPRANO project aims at solving these problems and prepare the next generation of verification-oriented solvers by gathering experts from academia and industry. We will design a new framework for the cooperation of solvers, focused on model generation and borrowing principles from SMT (current standard) and CP (well-known in optimisation). These ideas will be implemented in an open-source platform, with regular evaluations from the industrial partners.

8.2.2. BGLE / LEOC

8.2.2.1. BGLE2 CONNEXION

- Coordinator: EDF
- Other partners: Atos WorldGrid, Rolls-Royce Civil Nuclear, Corys TESS, Esterel Technologies, All4Tec, Predict, CEA, Inria, CNRS / CRAN, ENS Cachan, LIG, Telecom ParisTech
- Dates: 2012-2016
- Abstract: The cluster CONNEXION (*digital command CONntrol for Nuclear EXport and renovatION*) aims to propose and validate an innovative architecture platforms suitable control systems for nuclear power plants in France and abroad. In this project the Triskell team investigates methods and tools to (i) automatically analyze and compare regulatory requirements evolutions and geographical differences; (ii) automatically generate test cases for critical interactive systems.

8.2.2.2. LEOC CLARITY

- Coordinator: Obéo
- Other partners: AIRBUS, Airbus Defence and Space, All4tec, ALTRAN Technologies, AREVA, Artal, C.E.S.A.M.E.S., Eclipse Foundation Europe, Inria Sophia Antipolis Méditerranée, PRFC, Scilab Enterprises, Thales Global Services, Thales Alenia Space, Thales Research & Technology, Thales Systèmes Aéroportés, Université de Rennes 1.
- Dates: 2014-2017
- Abstract: The CLARITY project aims to establish an international dimension ecosystem around Melody/Capella modeling workbench for systems engineering (MBSE) and engineering architectures (system, software, hardware).

8.2.2.3. Occiware

- Coordinator: Open Wide
- Open Wide, ActiveEon SA, CSRT - Cloud Systèmes Réseaux et Télécoms, Institut Mines-Télécom/Télécom SudParis, Inria, Linagora, Obeo, OW2 Consortium, Pôle Numérique, Université Joseph Fourier,
- Dates: 2014-2017
- Abstract: The Occiware project aims to establish a formal and equipped framework for the management of all cloud resource based on the OCCI standard.

8.2.3. DGA

8.2.3.1. DGA-RAPID MOTIV

- Coordinator: InPixal
- Other partners: Bertin, DGA, Inria
- Dates: 2012-2014
- Abstract: This project investigates innovative software test generation and management solutions to handle the very high degrees of variability in video processing algorithmic chains. The objective is to provide systematic criteria to qualify the testing activity when developing video processing software and to tailor these criteria to the variability dimensions that emerge in the context of visible images.

8.2.3.2. DGA FPML

- Coordinator: DGA
- Partners: DGA MI, Inria
- Dates: 2014-2016
- Abstract: in the context of this project, DGA-MI and the Inria team DiverSE explore the existing approaches to ease the development of formal specifications of domain-Specific Languages (DSLs) dedicated to paquet filtering, while guaranteeing expressiveness, precision and safety. In the long term, this work is part of the trend to provide to DGA-MI and its partners a tooling to design and develop formal DSLs which ease the use while ensuring a high level of reasoning.

8.3. European Initiatives

8.3.1. FP7 & H2020 Projects

8.3.1.1. FP7 FET STREP DIVERSIFY

- Coordinator: Inria (DIVERSE)
- Other partners: SINTEF, Université de Rennes 1, Trinity College Dublin
- Dates: 2013-2016
- Abstract: DIVERSIFY explores diversity as the foundation for a novel software design principle and increased adaptive capacities in CASs. Higher levels of diversity in the system provide a pool of software solutions that can eventually be used to adapt to unforeseen situations at design time. The scientific development of DIVERSIFY is based on a strong analogy with ecological systems, biodiversity, and evolutionary ecology. DIVERSIFY brings together researchers from the domains of software-intensive distributed systems and ecology in order to translate ecological concepts and processes into software design principles.

8.3.1.2. FP7 NoE NESSoS

- Coordinator: CNR - Consiglio Nazionale delle Ricerche (Italy)
- Others partners: ATOS (Spain), ETH (Switzerland), Katholieke Universiteit Leuven (Belgium), Ludwig-Maximilians-Universitaet Muenchen (Germany), IMDEA (Spain), Inria (France), University of Duisburg-Essen (Germany), University of Malaga (Spain), University of Trento (Italy), SIEMENS (Germany), SINTEF (Norway)
- Dates: 2010-2014
- Abstract: The Network of Excellence on Engineering Secure Future Internet Software Services and Systems (NESSoS) aims at constituting and integrating a long lasting research community on engineering secure software-based services and systems. In light of the unique security requirements the Future Internet will expose, new results will be achieved by means of an integrated research, as to improve the necessary assurance level and to address risk and cost during the software development cycle in order to prioritize and manage investments. NESSoS will also impact training and education activities in Europe to grow a new generation of skilled researchers and practitioners in the area. NESSoS will collaborate with industrial stakeholders to improve the industry best practices and support a rapid growth of software-based service systems in the Future Internet.

  Three Inria EPIs are involved in NeSSoS: ARLES, CASSIS and Triskell. Triskell leads the research workpackage on design and architecture for secured future internet applications.

8.3.1.3. FP7 Marie-Curie Relate

- Coordinator: Karlsruhe Institute of Technology
- Other partners: Université de Rennes, IRISA (France); King's College (UK); South East European Research Center, SEERC (Greece); Charles University (Czech Republic); CAS Software (Germany); Singular Logic (Greece)
- Dates: 2011-2015
- Abstract: The RELATE Initial Training Network aims to establish a network of international academic and industrial partners for a joint research training effort in the area of engineering and provisioning service-based cloud applications. The training is intended to not only shape high-level academic researchers, but also educate next generation experts and innovators in the European software industry. Through an integrative and multidisciplinary research approach, RELATE aims to promote the advancement of the state of the art in the related areas of model-driven engineering and formal methods, service-based mash-ups and application integration, security, performance, and trust in service-based cloud applications, and quality management and business model innovation.

8.3.1.4. FP7 STREP HEADS

- Coordinator: SINTEF
- Other partners: Inria, Software AG, ATC, Tellu, eZmonitoring
- Dates: 2013-2016
- Abstract: The idea of the HEADS project is to leverage model-driven software engineering and generative programming techniques to provide a new integrated software engineering approach which allow advanced exploitation the full range of diversity and specificity of the future computing continuum. The goal is to empower the software and services industry to better take advantage of the opportunities of the future computing continuum and to effectively provide new innovative services that are seamlessly integrated to the physical world making them more pervasive, more robust, more reactive and closer (physically, socially, emotionally, etc.) to their users. We denote such services HD-services. HD-services (Heterogeneous and Distributed services) characterize the class of services or applications within the Future Internet whose logic and value emerges from a set of communicating software components distributed on a heterogeneous computing continuum from clouds to mobile devices, sensors and/or smart-objects.

8.3.2. Collaborations in European Programs, except FP7 & H2020

8.3.2.1. ICT COST Action MPM4CPS (IC1404)

- Chair of the Action: Prof Hans Vangheluwe (BE)
- Dates: 2014-2018
- Abstract: Truly complex, designed systems, known as Cyber Physical Systems (CPS), are emerging that integrate physical, software, and network aspects. To date, no unifying theory nor systematic design methods, techniques and tools exist for such systems. Individual (mechanical, electrical, network or software) engineering disciplines only offer partial solutions. Multi-paradigm Modelling (MPM) proposes to model every part and aspect of a system explicitly, at the most appropriate level(s) of abstraction, using the most appropriate modelling formalism(s). Modelling languages' engineering, including model transformation, and the study of their semantics, are used to realize MPM. MPM is seen as an effective answer to the challenges of designing CPS. This COST Action promotes the sharing of foundations, techniques and tools, and provide educational resources, to both academia and industry. This is achieved by bringing together and disseminating knowledge and experiments on CPS problems and MPM solutions.

8.3.3. Industry-driven EU projects

8.3.3.1. ITEA MERGE

- Coordinator: Thales Research and Technology
- Other partners: Thales Global Services, Thales Communications and Security, OBEO, ALL4TEC, Onera, Inria, Université Paris VI, Codenomicon, STUK - Radiation and Nuclear Safety Authority, POHTOnSense Oy, University of Oulu, University of Jyvaskyla, Space Applications Services NV, Melexis, E2S, Katholieke Universiteit Leuven
- Dates: 2012-2015
- Abstract: MERgE stands for "Multi-Concerns Interactions System Engineering". Within the "Engineering support" theme of ITEA2 roadmap, the purpose of this project is to develop and demonstrate innovative concepts and design tools addressing in combination the "Safety" and "Security" concerns, targeting the elaboration of effective architectural solutions. MERgE will provide tools and solutions for combining safety and security concerns in systems development in a holistic way. It will provide academically solid and practice proven solutions and models for system developers and system owners to tackle the challenges of designing seamless optimal cost effective safe and secure solutions conformant to the model driven engineering paradigm. This will be done by tightly integrating the following paradigms: requirement engineering, safety, security and risk management in an over-all design process which is supported by adequate tools and methods. MERgE aims to

bring a system engineering solution for Combined Safe & Secure system design. The main technical innovation of the project is the application of state of the art design tools tailorisation capabilities and "multi concern engineering" core technologies to the issue of interactions of "Safety" and "Security" concerns as well as other concerns like "Performance" or "Timing" in the design process.

8.3.4. Collaborations with Major European Organizations

SINTEF, ICT (Norway): Model-driven systems development for the construction of distributed, heterogeneous applications. We collaborate since 2008 and are currently in two FP7 projects together.

Université du Luxembourg, (Luxembourg): Models@runtime for dynamic adaptation and multi-objective elasticity in cloud management; model-driven development.

Open University (UK): models@runtime for the Internet of Things.

8.4. International Initiatives

8.4.1. Inria International Partners

8.4.1.1. Declared Inria International Partners

8.4.1.1.1. Inria International Chair

Prof. Robert B. France [11] was granted by an Inria international chair for the period 2013-2017. Prof. France collaborate intensively with many members of DIVERSE on various joint work, e.g., the Familiar project and the GEMOC initiative. The Inria International Chair allows Prof. France to visit once a year the team along the period.

8.4.1.2. Informal International Partners

- Université de Montréal (Canada)
- McGill University (Canada)
- University of Alabama (USA)

8.4.2. International initiative GEMOC

The GEMOC initiative (cf. http://www.gemoc.org) is an open and international initiative launched in 2013 that coordinate research partners worldwide to develop breakthrough software language engineering (SLE) approaches that support global software engineering through the use of multiple domain-specific languages. GEMOC members aim to provide effective SLE solutions to problems associated with the design and implementation of collaborative, interoperable and composable modeling languages.

The GEMOC initiative aims to provide a framework that facilitates collaborative work on the challenges of using of multiple domain-specific languages in software development projects. The framework consists of mechanisms for coordinating the work of members, and for disseminating research results and other related information on GEMOC activities. The framework also provides the required infrastructure for sharing artifacts produced by members, including publications, case studies, and tools.

The governance of the GEMOC initiative is ensured by the Advisory Board. The role of the Advisory Board is to coordinate the GEMOC work and to ensure proper dissemination of work products and information about GEMOC events (e.g., meetings, workshops).

Benoit Combemale is the co-founder and currently acts as principal coordinator of the GEMOC initiative. Benoit Combemale and Jean-Marc Jézéquel are part of the Advisory Board, and 9 DIVERSE members are part of the GEMOC initiative.

[11] Colorado State University, USA. See. http://www.cs.colostate.edu/~france/

8.5. International Research Visitors

8.5.1. Visits of International Scientists

8.5.1.1. Internships

- Victor Vranceanu: Inria, Intern, from Mar 2014 until Jul 2014
- Eric Manzi: Inria, Intern, from Jun 2014 until Aug 2014

9. Dissemination

9.1. Promoting Scientific Activities

9.1.1. Scientific events organisation

9.1.1.1. general chair, scientific chair

Benoit Combemale:

- chair for the program committee of SLE'14, the 7th International Conference on Software Language Engineering (SLE), Sweden, October 2014.
- co-chair of the demonstration track of the 17th International Conference on Model Driven Engineering Languages and Systems (MODELS), Spain, October, 2014.
- co-chair of the workshop GEMOC, co-located with the 17th International Conference on Model Driven Engineering Languages and Systems (MODELS), Spain, October, 2014.
- co-chair of the Dagstuhl seminar #14412 on *Globalizing Domain-Specific Languages*, Germany, October, 2014.
- co-chair of MDD4DRES 2014, the International School on Model-Driven Development for Distributed Realtime Embedded Systems, France, September, 2014.

Benoit Baudry:

- Co-chair of the PC, New Ideas and Emerging Results (NIER) track at ICSE'14
- Chair of the doctoral symposium, International Conference on Model Driven Engineering, Languages and Systems (MODELS)
- Member of the Steering committee, International Conference on Model Driven Engineering, Languages and Systems (MODELS)

Jean-Marc Jézéquel:

- co-chair of the Dagstuhl seminar #14412 on *Globalizing Domain-Specific Languages*, Germany, October, 2014.

9.1.1.2. member of the organizing committee

Mathieu Acher has organized the SPLTea workshop (co-located with the Software Product Line Conference) and Journée ligne de produits (7th edition) in Paris (Sorbonne).

9.1.2. Scientific events selection

9.1.2.1. responsable of the conference program committee

Jean-Marc Jézéquel has been a member of the program board:

- MODELS 2014, The 17th International Conference on Model Driven Engineering Languages and Systems Valencia, Spain, October 2014 (PB)
- ICSE 2014 The 36th International Conference on Software Engineering, Hyderabad, India, June 2014 (PB)

Benoit Baudry:

- Member of the Steering committee, International Conference on Model Driven Engineering, Languages and Systems (MODELS)

9.1.2.2. member of the conference program committee

Benoit Baudry:

- International Conference on Automated Software Engineering (ASE)
- International Conference on Model Driven Engineering, Languages and Systems (MODELS)
- International Working Conference on Requirements Engineering: Foundation for Software Quality (REFSQ)

Mathieu Acher:

- ICSE'15 (NIER) : 37th International Conference on Software Engineering - New Ideas and Emerging Results (NIER)
- ICSE'14 (NIER) : 36th International Conference on Software Engineering - New Ideas and Emerging Results (NIER)
- ICSE'14 (SEIP) : 36th International Conference on Software Engineering - Software Engineering In Practice (SEIP)
- SPLC'14 : 18th International Software Product Line Conference
- SEAA'14 : Euromicro Conference series on Software Engineering and Advanced Applications
- FOSD'14 : Sixth International Workshop on Feature-Oriented Software Development (at ASE)
- SPLTea : First International Workshop on Software Product Line Teaching
- REVE'14 : 2nd International Workshop on Reverse Variability Engineering (at SPLC)
- JLDP'14 : Journee Lignes de Produits (Luxembourg)

Olivier Barais:

- HPCC'14, The 16th IEEE International Conference on High Performance Computing and Communications, Paris, France
- SEAA'14, The 40th Euromicro Conference on Software Engineering and Advanced Applications (SEAA 2014) Verona, Italy 2014
- SAC'14, The 29th Symposium On Applied Computing Gyeongju, Korea.
- SPLat 2014, The 1st International Workshop on Software Product Line Analysis Tools, Convitto della Calza, sala Franciabigio
- VAO'14, The 2nd Workshop on View-Based, Aspect-Oriented and Orthographic Software Modelling, York, United Kingdom
- WEITICE'2014, the 23rd IEEE International Conference on Collaboration Technologies and Infrastructure University Campus Parma, Italy
- CIEL'14, the 3rd French Conference in Software Engineering. CNAM, Paris,
- CAL'14, the 5th French Conference in Software Architecture. Paris,

Arnaud Blouin:

- IEEE International Workshop on User Centered Design and Adaptive Systems (UCDAS), 2014

Johann Bourcier:

- PECCS 2013

Benoit Combemale:

- APSEC'14, The 21st Asia-Pacific Software Engineering Conference, Korea, 2014.
- ICMT'14, The 7th International Conference on Model Transformation, UK, 2014.
- ECMFA'14, The 10th European Conference on Modelling Foundations and Applications, UK, 2014.
- VOLT workshop on Verification Of ModeL Transformation (VOLT), at STAF'14, UK, 2014.
- CSP Trask at the WETICE'14, Italy, 2014.

Jean-Marc Jézéquel:

- Formal Demonstrations of ICSE 2014 The 36nd International Conference on Software Engineering, Hyderabad, India, June 1-7, 2014
- VaMoS 2014 The 8th International Workshop on Variability Modelling of Software-intensive Systems January 22-24, 2014 - Nice, France
- SLE 2014 7th International Conference on Software Language Engineering (SLE) September 15-16, 2014 Västerås, Sweden
- GPCE 2014 13th International Conference on Generative Programming: Concepts & Experiences September 15-16, 2014 Västerås, Sweden

Noël Plouzeau:

- CBSE 2014, Component Based Software Engineering, Lille, France, june 2014.

9.1.2.3. reviewer

Arnaud Blouin:

- EICS'14, ACM SIGCHI symposium on Engineering interactive computing systems.
- NordiCHI'14, Nordic Conference on Human-Computer Interaction.
- IHM'14, French speaking conference on human-computer interaction.
- ASPEC'14, Asia-Pacific Software Engineering Conference.

Benoit Combemale: external reviewer for ICSE'13 (International Conference on Software Engineering).

9.1.3. Journal

9.1.3.1. member of the editorial board

Benoit Baudry is an Associate Editor of the following journal:

- Journal of Systems and Software (JSS)
- Journal on Software and System Modeling (SoSYM)

Jean-Marc Jézéquel is Associate Editor-in-Chief of

- Journal on Software and System Modeling (SoSYM)

Jean-Marc Jézéquel is an Associate Editor of the following journal:

- IEEE Computer
- Journal of Systems and Software (JSS)
- Journal of Object Technology: JOT

Noël Plouzeau is chief editor of a special issue of the TSI French research journal, dedicated to software engineering.

Benoit Combemale is a member of the editorial board of the following journal:

- Elsevier's International Journal on Computer Languages, Systems and Structures (COMLAN)

9.1.3.2. reviewer

Mathieu Acher: reviewer for journals: ACM Computing Surveys, Information and Software Technology (IST), IEEE Software, Automated Software Engineering: An International Journal (ASE), Journal of Ambient Intelligence and Smart Environments (JAISE), Journal in Computers in Biology and Medicine (CBM), International Journal on Software Tools for Technology Transfer (STTT), Journal of Systems and Software (JSS), Journal of Software and Systems Modeling (SoSyM).

Olivier Barais: external reviewer for IEEE Transactions on Software Engineering, Journal of Software and Systems Modeling (SoSyM), Journal of Software System (JSS), ACM TaaS (Transactions on Autonomous and Adaptive Systems), STVR (Software Testing, Verification and Reliability).

Arnaud Blouin: external reviewer for SoSyM (Journal of Software and Systems Modeling).

Johann Bourcier: external reviewer for TaaS (Transactions on Autonomous and Adaptive Systems), and JSS (Journal of Systems and Software).

Benoit Combemale: external reviewer for IEEE Transactions on Software Engineering, Journal of Software and Systems Modeling (SoSyM).

Noël Plouzeau: external reviewer for JSS (Journal of Systems and Software).

9.2. Teaching - Supervision - Juries

9.2.1. Teaching

The DIVERSE team bears the bulk of the teaching on Software Engineering at the University of Rennes 1 and at INSA Rennes, at the levels M1 (Project Management, Object-Oriented Analysis and Design with UML, Design Patterns, Component Architectures and Frameworks, Validation & Verification, Human-Computer Interaction) and M2 (Model driven Engineering, Aspect-Oriented Software Development, Software Product Lines, Component Based Software Development, Validation & Verification, *etc.*).

Each of Jean-Marc Jézéquel, Noël Plouzeau, Olivier Barais, Johann Bourcier, Arnaud Blouin, and Mathieu Acher teaches about 200h in these domains, with Benoit Baudry and Benoit Combemale teaching about 50h, for a grand total of about 1300 hours, including several courses at ENSTB, Supelec and ENSAI Rennes.

Noël Plouzeau is the leader of the final year of the Master Pro in Computer Science at the University of Rennes 1.

Johann bourcier is co-manager of the Home-Automation option at the ESIR engineering school in Rennes.

Mathieu Acher is in charge of teaching services of ISTIC.

The DIVERSE team also receives several Master and summer trainees every year.

9.2.2. Supervision

HDR: Olivier Barais: *Utilisation de la modé lisation à l'exé cution : objectif, challenges et bé né fices*, 2014

HDR: Olivier Beaudoux: *Vers une programmation des systè mes interactifs centré e sur la spé cification de modè les exé cutables*, 2014

PhD: Joao Bosco Ferreira Filho, *Variability modeling in software-intensive systems*, thèse 2011-2014, O. Barais, B. Baudry and M. Acher

PhD: Emmanuelle Rouillé, *Processus Logiciels dirigés par les intentions*, thèse CIFRE avec SodiFrance, 2010-2014, J.-M. Jézéquel, B. Combemale and O. Barais

PhD : Julien Richard-Foy, *A DSL factory for modular Web oriented architecture*, thèse CIFRE avec Zenexity, 2011-2014, J.-M. Jézéquel and O. Barais

PhD : Hamza Samih, *Extending model-based testing with variability and security management*, thèse CIFRE avec All4Tec, 2011-2014, B. Baudry

PhD in progress : Erwan Bousse, *Intégration et combinaison des techniques de V&V dans un contexte d'ingénierie cystème*, thèse, 2012-2015, B. Baudry and B. Combemale

PhD in progress : Valéria Lelli, *On Testing Interactive Systems*, thèse 2012-2015, B. Baudry and A. Blouin

PhD in progress : Kwaku Yeboah-Antwi, *Runtime emergence of software diversity*, thèse 2013-2016, B. Baudry and O. Barais

PhD in progress : Sana Ben Nasr, *Modeling variability in regulatory requirements*, thèse 2013-2016, B. Baudry and M. Acher

PhD in progress : Guillaume Bécan, *Reverse Engineering and Evolution of Variability-intensive Software Systems*, thèse 2013-2016, B. Baudry and M. Acher

PhD in progress : Mohamed Boussaa, *An Architecture for Testing Large-Scale Dynamic Distributed Systems*, thèse 2013-2016, B. Baudry and O. Barais

PhD in progress : Thomas Degueule, *Next Generation of MDE Tooling Support*, thèse 2013-2016, O. Barais and A. Blouin

PhD in progress : David Mendez Acuna, *Variability in Modeling Languages*, thèse 2013-2016, B. Baudry and B. Combemale

PhD in progress : Ivan Paez Anaya, *Proactive Adaptation in Pervasive Environment*, thèse 2012-2015, J.Bourcier, N. Plouzeau and J.-M. Jézéquel

PhD in progress : Inti Gonzalez, *Ressources reservation in Pervasive Middleware*, thèse 2012-2015, J. Bourcier, O. Barais

PhD in progress : Jacky Bourgeois, *Automatic Synchronization of Energy Production and Consumption*, thèse 2012-2015 co-tutelle avec the Open University, J.Bourcier, B. Baudry and G. Kortuem

PhD in progress : Francisco Acosta, *Automatic Deployment and Reconfigurations in Internet of Things*, thèse 2013-2016 co-encadrement avec l'équipe ACES, J. Bourcier, F. Weis

PhD in progress : Pierre Laperdrix, *Secretless moving target against browser fingerprinting* , thèse 2014-2017, B. Baudry

PhD in progress : Johan Pelay, *Langage pour une programmation incrémentale de réseau*, thèse 2014-2017 co-encadrement avec le laboratoir B-COM, O.Barais, F. Guillemin

PhD in progress : Kévin Corre, *Modélisation de la confiance dans les services sociaux et conversationnels*, thèse 2014-2017 co-encadrement avec l'équipe AtanMod, O. Barais, G. Sunye

9.2.3. Juries

9.2.3.1. Benoit Combemale

was in the examination committee of the following PhD thesis:

- Emmanuelle Rouillé, April 2014, University of Rennes (Co-supervisor)
- Jean-Christophe Bach, September 2014, University of Nancy (Member)
- Faiez Zalila, December 2014, University of Toulouse (Member)

9.2.3.2. Jean-Marc Jézéquel

was in the examination committee of the following HDR:

- Olivier Beaudoux, August 2014, ESEO, President
- Rémi Cozot, September 2014, University of Rennes, President
- Laurent Amsaleg, November 2014, University of Rennes, President
- Maud Marchal, November 2014, University of Rennes, President
- Olivier Barais, December 2014, University of Rennes, member

9.2.3.3. Jean-Marc Jézéquel

was in the examination committee of the following PhD thesis:

- Emmanuelle Rouillé, April 2014, University of Rennes, Supervisor
- Cyrille Jegourel, November 2014, University of Rennes, President
- Minh Tu Ton That, October 2014, Université de Bretagne-Sud, President
- Julien Richard-Foy, December 2014, University of Rennes, Supervisor

9.2.3.4. Olivier Barais

was in the examination committee of the following PhD thesis:

- Emmanuelle Rouillé, April 2014, University of Rennes, Supervisor
- Julien Richard-Foy, December 2014, University of Rennes, Supervisor
- Bosco Ferreira, December 2014, University of Rennes, Supervisor

9.2.3.5. Benoit Baudry

was in the examination committee of the following HDR:

- Olivier Barais, December 2014, University of Rennes, member

9.2.3.6. Benoit Baudry

was in the examination committee of the following PhD thesis:

- Hamza Samih, December 2014, University of Rennes, Supervisor
- Bosco Ferreira, December 2014, University of Rennes, Supervisor
- Jérémie Tatibouet, Ocotber 2014, University of Paris-Sud, Referee
- Boris Baldassari, June 2014, University of Lille, Referee

9.2.3.7. Mathieu Acher

was in the examination committee of the following PhD thesis:

- Ebrahim Abbasi, February 2014, University of Namur (Belgium), Referee
- Bosco Ferreira, December 2014, University of Rennes

9.2.3.8. Johann Bourcier

was in the examination committee of the following PhD thesis:

- Koutheir Attouchi, July 2014, University of Paris 6,

9.3. Popularization

Our recent work on diversity against browser fingerprinting and the associated web site [12] have been demonstrated in two open source forums (OWF'14 [13] and FOSSA'14 [14]. Following this, our work has been mentioned in general-audience press. In particular the Clubic magazine published a complete article about our work [15], which was followed by 6000 unique visits on our web site in the following 24 hours. The work was also mentioned in Le Monde [16].

[12] https://amiunique.org/
[13] http://www.openworldforum.paris/
[14] https://fossa.inria.fr/
[15] http://pro.clubic.com/webmarketing/publicite-en-ligne/actualite-742853-fingerprinting-cookies.html
[16] http://abonnes.lemonde.fr/sciences/article/2014/11/24/le-fossa-bazar-techno-participatif_4528568_1650684.html

10. Bibliography

Major publications by the team in recent years

[1] M. ACHER, P. COLLET, P. LAHIRE, R. FRANCE. *FAMILIAR: A Domain-Specific Language for Large Scale Management of Feature Models*, in "Science of Computer Programming", June 2013, vol. 78, n^{o} 6, pp. 657 - 681 [*DOI :* 10.1016/J.SCICO.2012.12.004], https://hal.inria.fr/hal-00767175

[2] B. BAUDRY, S. GHOSH, F. FLEUREY, R. FRANCE, Y. LE TRAON, J.-M. MOTTU. *Barriers to Systematic Model Transformation Testing*, in "Communications of the ACM journal", Jun 2010, vol. 53, n^{o} 6, 10 p.

[3] J. BOURCIER, A. DIACONESCU, P. LALANDA, JULIE A. MCCANN. *AutoHome: an Autonomic Management Framework for Pervasive Home Applications*, in "ACM Transactions on Autonomous and Adaptive Systems", February 2011, vol. 6, n^{o} 1, https://hal.inria.fr/inria-00554197

[4] B. COMBEMALE, J. DEANTONI, B. BAUDRY, R. FRANCE, J.-M. JÉZÉQUEL, J. GRAY. *Globalizing Modeling Languages*, in "Computer", June 2014, pp. 10-13, https://hal.inria.fr/hal-00994551

[5] F. FOUQUET, E. DAUBERT, N. PLOUZEAU, O. BARAIS, J. BOURCIER, A. BLOUIN. *Kevoree : une approche model@runtime pour les systèmes ubiquitaires*, in "UbiMob2012", Anglet, France, June 2012, https://hal.inria.fr/hal-00714557

[6] C. JEANNERET, M. GLINZ, B. BAUDRY. *Estimating Footprints of Model Operations*, in "International Conference on Software Engineering", Honolulu, United States, May 2011, https://hal.inria.fr/hal-00641091

[7] J.-M. JÉZÉQUEL, B. COMBEMALE, O. BARAIS, M. MONPERRUS, F. FOUQUET. *Mashup of Meta-Languages and its Implementation in the Kermeta Language Workbench*, in "Software and Systems Modeling", 2013, https://hal.inria.fr/hal-00829839

[8] B. MORIN, O. BARAIS, J.-M. JÉZÉQUEL, F. FLEUREY, A. SOLBERG. *Models at Runtime to Support Dynamic Adaptation*, in "IEEE Computer", October 2009, pp. 46-53, http://www.irisa.fr/triskell/publis/2009/Morin09f.pdf

Publications of the year

Doctoral Dissertations and Habilitation Theses

[9] O. BARAIS. *Utilisation de la modélisation à l'exécution : objectif, challenges et bénéfices*, Université de Rennes 1, December 2014, Habilitation à diriger des recherches, https://hal.inria.fr/tel-01096952

[10] J. B. F. FILHO. *Leveraging model-based product lines for systems engineering*, Université Rennes 1, December 2014, https://hal.inria.fr/tel-01092054

[11] J. RICHARD-FOY. *Web applications engineering: reduce the complexity without loosing control*, Université de Rennes 1, France, December 2014, https://hal.inria.fr/tel-01087372

[12] E. ROUILLÉ. *Variability management and automation of software development processes*, Université Rennes 1, April 2014, https://tel.archives-ouvertes.fr/tel-01061129

[13] H. SAMIH. *Model-based Testing applied to product line*, Université de Rennes 1, December 2014, https://hal.inria.fr/tel-01092342

Articles in International Peer-Reviewed Journals

[14] M. E. ALFÉREZ, R. BONIFÁCIO, L. TEIXEIRA, P. ACCIOLY, U. KULESZA, A. MOREIRA, J. ARAUJO, P. BORBA. *Evaluating scenario-based SPL requirements approaches: the case for modularity, stability and expressiveness*, in "Requirements Engineering", November 2014, vol. 19, n° 4, pp. 355 - 376 [*DOI :* 10.1007/s00766-013-0184-5], https://hal.inria.fr/hal-01088537

[15] M. E. ALFÉREZ, R. E. LOPEZ-HERREJON, A. MOREIRA, V. AMARAL, A. EGYED. *Consistency Checking in Early Software Product Line Specifications -The VCC Approach*, in "Journal of Universal Computer Science", May 2014, vol. 20, n° 5, pp. 640 - 665, https://hal.inria.fr/hal-01088464

[16] S. ALLIER, O. BARAIS, B. BAUDRY, J. BOURCIER, E. DAUBERT, F. FLEUREY, M. MONPERRUS, H. SONG, M. TRICOIRE. *Multi-tier diversification in Web-based software applications*, in "IEEE Software", 2014, 7 p. , https://hal.archives-ouvertes.fr/hal-01089268

[17] M. AMRANI, B. COMBEMALE, L. LÚCIO, G. SELIM, J. DINGEL, Y. LE TRAON, H. VANGHELUWE, J. R. CORDY. *Formal Verification Techniques for Model Transformations: A Tridimensional Classification*, in "Journal of Object Technology", 2014, pp. 1 - 43 [*DOI :* 10.5381/jot.201Y.VV.N.aN], https://hal.inria.fr/hal-01083759

[18] V. ARANEGA, J.-M. MOTTU, A. ETIEN, T. DEGUEULE, B. BAUDRY, J.-L. DEKEYSER. *Towards an Automation of the Mutation Analysis Dedicated to Model Transformation*, in "Software Testing, Verification and Reliability", April 2014 [*DOI :* 10.1002/stvr.1532], https://hal.inria.fr/hal-00988164

[19] G. BÉCAN, M. ACHER, B. BAUDRY, S. BEN NASR. *Breathing Ontological Knowledge Into Feature Model Synthesis: An Empirical Study*, in "Empirical Software Engineering", 2015, 51 p. [*DOI :* 10.1007/s10664-014-9357-1], https://hal.inria.fr/hal-01096969

[20] B. COMBEMALE, J. DEANTONI, B. BAUDRY, R. FRANCE, J.-M. JÉZÉQUEL, J. GRAY. *Globalizing Modeling Languages*, in "Computer", June 2014, pp. 10-13, https://hal.inria.fr/hal-00994551

[21] D. EFSTATHIOU, P. MCBURNEY, S. ZSCHALER, J. BOURCIER. *Efficient Multi-Objective Optimisation of Service Compositions in Mobile Ad hoc Networks Using Lightweight Surrogate Models*, in "Journal of Universal Computer Science", August 2014, vol. 20, n° 8, 20 p. , https://hal.inria.fr/hal-01090552

[22] J. B. FERREIRA FILHO, O. BARAIS, M. ACHER, J. LE NOIR, A. LEGAY, B. BAUDRY. *Generating Counterexamples of Model-based Software Product Lines*, in "Software Tools for Technology Transfer (STTT)", July 2014, https://hal.inria.fr/hal-01026581

[23] G. SUNYÉ, E. CUNHA DE ALMEIDA, Y. LE TRAON, B. BAUDRY, J.-M. JÉZÉQUEL. *Model-Based Testing of Global Properties on Large-Scale Distributed Systems*, in "Information and Software Technology", 2014, https://hal.inria.fr/hal-00942576

International Conferences with Proceedings

[24] E. K. ABBASI, M. ACHER, P. HEYMANS, A. CLEVE. *Reverse Engineering Web Configurators*, in "17th European Conference on Software Maintenance and Reengineering (CSMR)", Antwerp, Belgium, IEEE, February 2014, https://hal.inria.fr/hal-00913139

[25] M. ACHER, M. E. ALFÉREZ, J. A. GALINDO, P. ROMENTEAU, B. BAUDRY. *ViViD: A Variability-Based Tool for Synthesizing Video Sequences*, in "SPLC'14 (tool demonstration track)", Florence, Italy, September 2014, https://hal.inria.fr/hal-01020933

[26] M. ACHER, B. BAUDRY, O. BARAIS, J.-M. JÉZÉQUEL. *Customization and 3D Printing: A Challenging Playground for Software Product Lines*, in "18th International Software Product Line Conference", Florence, Italy, July 2014, https://hal.inria.fr/hal-01018937

[27] M. ACHER, B. COMBEMALE, P. COLLET. *Metamorphic Domain-Specific Languages: A Journey Into the Shapes of a Language*, in "Onward! Essays", Portland, United States, ACM, September 2014, pp. 243–253 [*DOI :* 10.1145/2661136.2661159], https://hal.inria.fr/hal-01061576

[28] M. ACHER, R. E. LOPEZ-HERREJON, R. RABISER. *A Survey on Teaching of Software Product Lines*, in "Eight International Workshop on Variability Modelling of Software-Intensive Systems (VaMoS'14)", Nice, France, ACM, January 2014, pp. 1-8 [*DOI :* 10.1145/2556624.2556629], https://hal.inria.fr/hal-00916746

[29] M. ACHER, R. E. LOPEZ-HERREJON, R. RABISER. *SPLTea 2014: First International Workshop on Software Product Line Teaching*, in "SPLC - 18th International Software Product Line Conference", Florence, Italy, September 2014, https://hal.inria.fr/hal-01024990

[30] F. J. ACOSTA PADILLA, F. WEIS, J. BOURCIER. *Towards a Model@runtime Middleware for Cyber Physical Systems*, in "Proceedings of the 9th Workshop on Middleware for Next Generation Internet Computing", Bordeaux, France, December 2014, https://hal.inria.fr/hal-01090269

[31] R. AL ALI, I. GEROSTATHOPOULOS, I. GONZALEZ-HERRERA, A. JUAN-VERDEJO, M. KIT, B. SURAJBALI. *An Architecture-Based Approach for Compute-Intensive Pervasive Systems in Dynamic Environments*, in "International Workshop on Hot TopiCS in Cloud Cloud service Scalability (HotTopiCS 2014)", Dublin, Ireland, March 2014, https://hal.inria.fr/hal-01091541

[32] B. BAUDRY, S. ALLIER, M. MONPERRUS. *Tailored Source Code Transformations to Synthesize Computationally Diverse Program Variants*, in "ISSTA", United States, 2014, pp. 149-159 [*DOI :* 10.1145/2610384.2610415], https://hal.archives-ouvertes.fr/hal-00938855

[33] B. BAUDRY, M. MONPERRUS, C. MONY, F. CHAUVEL, F. FLEUREY, S. CLARKE. *DIVERSIFY - Ecology-inspired software evolution for diversity emergence*, in "CSMR", Antwerp, Belgium, IEEE, February 2014, pp. 395-398, https://hal.inria.fr/hal-00916281

[34] S. BEN NASR, N. SANNIER, M. ACHER, B. BAUDRY. *Moving Toward Product Line Engineering in a Nuclear Industry Consortium*, in "18th International Software Product Line Conference (SPLC'2014)", Florence, Italy, September 2014, https://hal.inria.fr/hal-01019537

[35] M. BIAZZINI, B. BAUDRY. *"May the fork be with you": novel metrics to analyze collaboration on GitHub*, in "Proceedings of the 5th International Workshop on Emerging Trends in Software Metrics (WETSoM 2014)", Hyderabad, India, June 2014, pp. 37 - 43 [*DOI :* 10.1145/2593868.2593875], https://hal.inria.fr/hal-01085400

[36] M. BIAZZINI, M. MONPERRUS, B. BAUDRY. *On Analyzing the Topology of Commit Histories in Decentralized Version Control Systems*, in "ICSME 2014", Victoria, Canada, 2014, pp. 261-270, pages: 10, https://hal.archives-ouvertes.fr/hal-01063789

[37] A. BLOUIN, N. MOHA, B. BAUDRY, H. SAHRAOUI. *Slicing-based Techniques for Visualizing Large Metamodels*, in "IEEE Working Conference on Software Visualization (VISSOFT 2014)", Victoria, Canada, September 2014, https://hal.inria.fr/hal-01056217

[38] J. BOURGEOIS, J. VAN DER LINDEN, G. KORTUEM, B. A. PRICE, C. RIMMER. *Conversations with my Washing Machine: An in-the-wild Study of Demand Shifting with Self-generated Energy*, in "2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp 2014)", Seattle, WA, United States, September 2014 [*DOI :* 10.1145/2632048.2632106], https://hal.inria.fr/hal-01090641

[39] J. BOURGEOIS, J. VAN DER LINDEN, G. KORTUEM, C. RIMMER. *Using Participatory Data Analysis to Understand Social Constraints and Opportunities of Electricity Demand-Shifting*, in "2nd International Conference on ICT for Sustainability (ICT4S 2014)", Stockholm, Sweden, August 2014 [*DOI :* 10.2991/ICT4S-14.2014.49], https://hal.inria.fr/hal-01090643

[40] E. BOUSSE, B. COMBEMALE, B. BAUDRY. *Scalable Armies of Model Clones through Data Sharing*, in "Model Driven Engineering Languages and Systems, 17th International Conference - MODELS", Valencia, Spain, September 2014, https://hal.inria.fr/hal-01023681

[41] E. BOUSSE, B. COMBEMALE, B. BAUDRY. *Towards Scalable Multidimensional Execution Traces for xDSMLs*, in "11th Workshop on Model Design, Verification and Validation Integrating Verification and Validation in MDE (MoDeVVa)", Valencia, Spain, September 2014, https://hal.inria.fr/hal-01061740

[42] G. BÉCAN, S. BEN NASR, M. ACHER, B. BAUDRY. *WebFML: Synthesizing Feature Models Everywhere*, in "SPLC - 18th International Software Product Line Conference", Florence, Italy, September 2014, https://hal.inria.fr/hal-01022912

[43] G. BÉCAN, N. SANNIER, M. ACHER, O. BARAIS, A. BLOUIN, B. BAUDRY. *Automating the Formalization of Product Comparison Matrices*, in "ASE - 29th IEEE/ACM International Conference on Automated Software Engineering", Västerås, Sweden, September 2014 [*DOI :* 10.1145/2642937.2643000], https://hal.inria.fr/hal-01058440

[44] J. DEANTONI, P. ISSA DIALLO, C. TEODOROV, J. CHAMPEAU, B. COMBEMALE. *Towards a Meta-Language for the Concurrency Concern in DSLs*, in "Design, Automation and Test in Europe Conference and Exhibition (DATE)", Grenoble, France, March 2015, https://hal.inria.fr/hal-01087442

[45] T. DUVAL, A. BLOUIN, J.-M. JÉZÉQUEL. *When Model Driven Engineering meets Virtual Reality: Feedback from Application to the Collaviz Framework*, in "Software Engineering and Architectures for Realtime Interactive Systems Working Group", Minnesota, United States, March 2014, https://hal.inria.fr/hal-00969072

[46] D. EFSTATHIOU, P. MCBURNEY, S. ZSCHALER, J. BOURCIER. *Surrogate-Assisted Optimisation of Composite Applications in Mobile Ad hoc Networks*, in "GECCO - Genetic and Evolutionary Computation Conference", Vancouver, Canada, July 2014, https://hal.inria.fr/hal-00983064

[47] D. EL KATEB, F. FOUQUET, J. BOURCIER, Y. LE TRAON. *Optimizing Multi-objective Evolutionary Algorithms to Enable Quality-Aware Software Provisioning*, in "The 14th International Conference on Quality

Software", Dallas, United States, October 2014, pp. 85 - 94 [*DOI :* 10.1109/QSIC.2014.44], https://hal.inria.fr/hal-01090246

[48] U. FAHRENBERG, M. ACHER, A. LEGAY, A. WĄSOWSKI. *Sound Merging and Differencing for Class Diagrams*, in "FASE 2014 : 17th International Conference on Fundamental Approaches to Software Engineering", Grenoble, France, S. GNESI, A. RENSINK (editors), LNCS : Fundamental Approaches to Software Engineering, Springer, April 2014, vol. 8411, pp. 63 - 78 [*DOI :* 10.1007/978-3-642-54804-8_5], https://hal.inria.fr/hal-01087323

[49] F. FOUQUET, G. NAIN, E. DAUBERT, J. BOURCIER, O. BARAIS, N. PLOUZEAU, B. MORIN. *Designing and Evolving Distributed Architecture using Kevoree*, in "QoSA '14 Proceedings of the 10th international ACM Sigsoft conference on Quality of software architectures", Lille, France, ACM New York, NY, USA ©2014, July 2014, pp. 147-148 [*DOI :* 10.1145/2602576.2611461], https://hal.inria.fr/hal-01096941

[50] J. A. GALINDO DUARTE, M. E. ALFÉREZ, M. ACHER, B. BAUDRY, D. BENAVIDES. *A Variability-Based Testing Approach for Synthesizing Video Sequences*, in "ISSTA '14: International Symposium on Software Testing and Analysis", San José, California, United States, July 2014, https://hal.inria.fr/hal-01003148

[51] I. GONZALEZ-HERRERA, J. BOURCIER, O. BARAIS, F. FOUQUET. *Designing resource-aware distriubted system based on system level containers*, in "Middleware Conference", Bordeaux, France, Middleware tutorial, December 2014, 2 p. , https://hal.inria.fr/hal-01090565

[52] I. GONZALEZ-HERRERA, J. BOURCIER, E. DAUBERT, W. RUDAMETKIN, O. BARAIS, F. FOUQUET, J.-M. JÉZÉQUEL. *Scapegoat: an Adaptive monitoring framework for Component-based systems*, in "Working IEEE/IFIP Conference on Software Architecture", Syndey, Australia, A. TANG (editor), IEEE/IFIP, April 2014, https://hal.inria.fr/hal-00983045

[53] T. HARTMANN, F. FOUQUET, G. NAIN, B. MORIN, J. KLEIN, O. BARAIS, Y. LE TRAON. *A Native Versioning Concept to Support Historized Models at Runtime*, in "ACM/IEEE 17th International Conference on Model Driven Engineering Languages and Systems (MODELS 2014)", Valencia, Spain, September 2014, pp. 252 - 268 [*DOI :* 10.1007/978-3-319-11653-2_16], https://hal.inria.fr/hal-01097020

[54] J.-M. JÉZÉQUEL. *Domain Specific Languages: From Craft to Engineering*, in "The 16th International Conference on Information Integration and Web-based Applications & Services", Hanoi, Vietnam, December 2014, https://hal.inria.fr/hal-01098012

[55] J.-M. JÉZÉQUEL. *Safely Reusing Model Transformations through Family Polymorphism*, in "8th System Analysis and Modelling Confrence (SAM2014)", Valencia, Spain, September 2014, https://hal.inria.fr/hal-01097975

[56] J.-M. JÉZÉQUEL, D. MENDEZ, T. DEGUEULE, B. COMBEMALE, O. BARAIS. *When Systems Engineering Meets Software Language Engineering*, in "CSD&M'14 - Complex Systems Design & Management", Paris, France, Springer, November 2014, https://hal.inria.fr/hal-01024166

[57] M. KEZADRI, M. PANTEL, B. COMBEMALE, X. THIRIOUX. *A formal framework to prove the correctness of model driven engineering composition operators*, in "ICFEM'14 - 16th International Conference on Formal Engineering Methods", Luxembourg, Springer, November 2014, https://hal.inria.fr/hal-01024067

[58] G. MUSSBACHER, D. AMYOT, R. BREU, J.-M. BRUEL, B. H. CHENG, P. COLLET, B. COMBEMALE, R. FRANCE, R. HELDAL, J. HILL, J. KIENZLE, M. SCHÖTTLE, F. STEIMANN, D. STIKKOLORUM, J. WHITTLE. *The Relevance of Model-Driven Engineering Thirty Years from Now*, in "Model-Driven Engineering Languages and Systems", Valencia, Spain, J. DINGEL, W. SCHULTE, I. RAMOS, S. ABRAHÃO, E. INSFRAN (editors), Model-Driven Engineering Languages and Systems, Springer International Publishing Switzerland, September 2014, vol. 8767, 18 p. [*DOI :* 10.1007/978-3-319-11653-2_12], https://hal.inria.fr/hal-01081848

[59] D. MÉNDEZ-ACUÑA. *Variability Management in Domain-Specific Languages*, in "Doctoral Symposium of 17th International Conference in Model-Driven Engineering Languages and Systems", Valencia, Spain, B. BAUDRY (editor), September 2014, https://hal.inria.fr/hal-01077834

[60] I. D. PAEZ ANAYA, V. SIMKO, J. BOURCIER, N. PLOUZEAU, J.-M. JÉZÉQUEL. *A Prediction-Driven Adaptation Approach for Self-Adaptive Sensor Networks*, in "9th International Symposium on Software Engineering for Adaptive and Self-Managing Systems", Hyderabad, India, N. BENCOMO (editor), IEEE/ACM, June 2014, https://hal.inria.fr/hal-00983046

[61] S. PINCHINAT, M. ACHER, D. VOJTISEK. *Towards Synthesis of Attack Trees for Supporting Computer-Aided Risk Analysis*, in "Workshop on Formal Methods in the Development of Software (co-located with SEFM)", Grenoble, France, September 2014, https://hal.inria.fr/hal-01064645

[62] J. RICHARD-FOY, O. BARAIS, J.-M. JÉZÉQUEL. *Using Path-Dependent Types to Build Type Safe JavaScript Foreign Function Interfaces*, in "ICWE - 14th International Conference on Web Engineering", Toulouse, France, July 2014, https://hal.inria.fr/hal-01026148

[63] H. SAMIH, M. ACHER, R. BOGUSCH, H. LE GUEN, B. BAUDRY. *Deriving Usage Model Variants for Model-based Testing: An Industrial Case Study*, in "19th International Conference on Engineering of Complex Computer Systems (ICECCS 2014)", Tianjin, China, IEEE, August 2014, https://hal.inria.fr/hal-01002099

[64] H. SAMIH, R. BOGUSCH. *MPLM – MaTeLo Product Line Manager*, in "18th International Software Product Line Conference (2014)", Florence, Italy, September 2014, https://hal.inria.fr/hal-01025159

[65] H. SAMIH, H. LE GUEN, R. BOGUSCH, M. ACHER, B. BAUDRY. *An Approach to Derive Usage Models Variants for Model-based Testing*, in "The 26th IFIP International Conference on Testing Software and Systems (2014)", Madrid, Spain, Springer, September 2014, https://hal.inria.fr/hal-01025124

[66] N. SANNIER, B. BAUDRY. *INCREMENT: A Mixed MDE-IR Approach for Regulatory Requirements Modeling and Analysis*, in "REFSQ'2014 - the 20th International Working Conference on Requirements Engineering: Foundation for Software Quality", Essen, Germany, C. SALINESI, I. VAN DE WEERD (editors), Springer, April 2014, https://hal.inria.fr/hal-00982065

[67] N. SANNIER, G. BÉCAN, M. ACHER, S. BEN NASR, B. BAUDRY. *Comparing or Configuring Products: Are We Getting the Right Ones?*, in "8th International Workshop on Variability Modelling of Software-intensive Systems", Nice, France, A. WASOWSKI, T. WEYER (editors), ACM, January 2014 [*DOI :* 10.1145/2556624.2556636], https://hal.inria.fr/hal-00927312

[68] E. VACCHI, W. CAZZOLA, B. COMBEMALE, M. ACHER. *Automating Variability Model Inference for Component-Based Language Implementations*, in "SPLC'14 - 18th International Software Product Line

Conference", Florence, Italy, P. HEYMANS, J. RUBIN (editors), ACM, September 2014, https://hal.inria.fr/hal-01023864

[69] D. VAN LANDUYT, S. OP DE BEECK, A. HOVSEPYAN, S. MICHIELS, W. JOOSEN, S. MEYNCKENS, G. DE JONG, O. BARAIS, M. ACHER. *Towards Managing Variability in the Safety Design of an Automotive Hall Effect Sensor*, in "18th International Software Product Line Conference", Florence, Italy, September 2014, https://hal.inria.fr/hal-01018938

[70] A. YOL, B. DELABARRE, A. DAME, J.-E. DARTOIS, E. MARCHAND. *Vision-based Absolute Localization for Unmanned Aerial Vehicles*, in "IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IROS'14", Chicago, United States, September 2014, https://hal.inria.fr/hal-01010140

National Conferences with Proceedings

[71] S. DUPUY-CHESSA, B. COMBEMALE, M.-P. GERVAIS, T. NODENOT, X. LE PALLEC, L. WOUTERS. *Vers une approche centrée humain pour la définition de langages de modélisation graphiques*, in "32ème congrès Inforsid'2014", Lyon, France, May 2014, pp. 79-94, https://hal.archives-ouvertes.fr/hal-01002994

Conferences without Proceedings

[72] M. ACHER, O. BARAIS, B. BAUDRY, A. BLOUIN, J. BOURCIER, B. COMBEMALE, J.-M. JÉZÉQUEL, N. PLOUZEAU. *Software Diversity: Challenges to handle the imposed, Opportunities to harness the chosen*, in "GDR GPL", Paris, France, June 2014, https://hal.inria.fr/hal-00980126

[73] G. BÉCAN, M. ACHER, J.-M. JÉZÉQUEL, T. MENGUY. *On the Variability Secrets of an Online Video Generator*, in "Variability Modelling of Software-intensive Systems", Hildesheim, Germany, January 2015, pp. 96 - 102 [*DOI :* 10.1145/2701319.2701328], https://hal.inria.fr/hal-01104797

[74] V. LELLI, A. BLOUIN, B. BAUDRY. *Classifying and Qualifying GUI Defects*, in "8th IEEE International Conference on Software Testing, Verification and Validation", Graz, Austria, April 2015, https://hal.inria.fr/hal-01114724

Scientific Books (or Scientific Book chapters)

[75] M. ACHER, P. COLLET, P. LAHIRE. *Architectures logicielles et variabilité multiple*, in "Architectures logicielles : principes, techniques et outils", Lavoisier, February 2014, https://hal.inria.fr/hal-01098109

[76] M. ACHER, P. COLLET, P. LAHIRE. *Software Architectures and Multiple Variability*, in "Software Architecture 2", ISTE Editions, May 2014, https://hal.inria.fr/hal-01098107

Books or Proceedings Editing

[77] B. COMBEMALE, J. DEANTONI, R. FRANCE (editors). *GEMOC 2014 2nd International Workshop on The Globalization of Modeling Languages*, CEUR-WS, September 2014, vol. 1236, 82 p. , https://hal.inria.fr/hal-01074602

[78] B. COMBEMALE, D. J. PEARCE, O. BARAIS, J. J. VINJU (editors). *Software Language Engineering*, SpringerVästerås, Sweden, 2014, n^{o} 8706, 353 p. , https://hal.inria.fr/hal-01110914

Research Reports

[79] M. E. ALFÉREZ, J. A. GALINDO, M. ACHER, B. BAUDRY. *Modeling Variability in the Video Domain: Language and Experience Report*, July 2014, n° RR-8576, https://hal.inria.fr/hal-01023159

[80] B. BAUDRY, M. MONPERRUS. *The Multiple Facets of Software Diversity: Recent Developments in Year 2000 and Beyond*, 2014, https://hal.archives-ouvertes.fr/hal-01067782

[81] J. DEANTONI, P. I. DIALLO, J. CHAMPEAU, B. COMBEMALE, C. TEODOROV. *Operational Semantics of the Model of Concurrency and Communication Language*, September 2014, n° RR-8584, 23 p. , https://hal.inria.fr/hal-01060601

[82] F. FOUQUET, G. NAIN, B. MORIN, E. DAUBERT, O. BARAIS, N. PLOUZEAU, J.-M. JÉZÉQUEL. *Kevoree Modeling Framework (KMF): Efficient modeling techniques for runtime use*, University of Luxembourg, May 2014, n° TR-SnT-2014-11, 24 p. , ISBN 978-2-87971-131-7, https://hal.inria.fr/hal-00996764

[83] D. MENDEZ, B. BAUDRY, M. MONPERRUS. *Analysis and Exploitation of Natural Software Diversity: The Case of API Usages*, Inria, 2014, https://hal.archives-ouvertes.fr/hal-01095501

Other Publications

[84] T. DEGUEULE, O. BARAIS, A. BLOUIN, B. COMBEMALE. *The K3 Model-Based Language Workbench*, July 2014, https://hal.inria.fr/hal-01025283

[85] T. DEGUEULE, B. COMBEMALE, O. BARAIS, A. BLOUIN, J.-M. JÉZÉQUEL. *Leveraging Family Polymorphism in MDE*, May 2014, https://hal.inria.fr/hal-00994541

[86] D. MÉNDEZ-ACUÑA, B. COMBEMALE, B. BAUDRY. *Variability Management in Domain-Specific Languages*, September 2014, International School of Model-Driven Development for Distributed Real-Time Embedded Systems (MDDRES), https://hal.inria.fr/hal-01077788

References in notes

[87] M. ACHER, A. CLEVE, P. COLLET, P. MERLE, L. DUCHIEN, P. LAHIRE. *Extraction and Evolution of Architectural Variability Models in Plugin-based Systems*, in "Software & Systems Modeling (SoSyM)", July 2013, 27 p. [*DOI :* 10.1007/s10270-013-0364-2], http://hal.inria.fr/hal-00859472

[88] M. ACHER, B. COMBEMALE, P. COLLET, O. BARAIS, P. LAHIRE, R. FRANCE. *Composing your Compositions of Variability Models*, in "ACM/IEEE 16th International Conference on Model Driven Engineering Languages and Systems (MODELS'13)", Miami, United States, September 2013, vol. Lecture Notes in Computer Science, 17 p. , http://hal.inria.fr/hal-00859473

[89] M. ACHER, P. HEYMANS, A. CLEVE, J.-L. HAINAUT, B. BAUDRY. *Support for Reverse Engineering and Maintaining Feature Models*, in "VaMoS'13 - Seventh International Workshop on Variability Modelling of Software-Intensive Systems", Pisa, Italy, ACM, January 2013, http://hal.inria.fr/hal-00766786

[90] A. ARCURI, L. C. BRIAND. *A practical guide for using statistical tests to assess randomized algorithms in software engineering*, in "ICSE", 2011, pp. 1-10

[91] A. AVIZIENIS. *The N-version approach to fault-tolerant software*, in "Software Engineering, IEEE Transactions on", 1985, n^o 12, pp. 1491–1501

[92] F. BACHMANN, L. BASS. *Managing variability in software architectures*, in "SIGSOFT Softw. Eng. Notes", 2001, vol. 26, n^o 3, pp. 126–132

[93] F. BALARIN, Y. WATANABE, H. HSIEH, L. LAVAGNO, C. PASSERONE, A. SANGIOVANNI-VINCENTELLI. *Metropolis: An integrated electronic system design environment*, in "Computer", 2003, vol. 36, n^o 4, pp. 45–52

[94] E. BANIASSAD, S. CLARKE. *Theme: an approach for aspect-oriented analysis and design*, in "26th International Conference on Software Engineering (ICSE)", 2004, pp. 158-167

[95] E. G. BARRANTES, D. H. ACKLEY, S. FORREST, D. STEFANOVIĆ. *Randomized instruction set emulation*, in "ACM Transactions on Information and System Security (TISSEC)", 2005, vol. 8, n^o 1, pp. 3–40

[96] D. BATORY, R. E. LOPEZ-HERREJON, J.-P. MARTIN. *Generating Product-Lines of Product-Families*, in "ASE '02: Automated software engineering", IEEE, 2002, pp. 81–92

[97] S. BECKER, H. KOZIOLEK, R. REUSSNER. *The Palladio component model for model-driven performance prediction*, in "Journal of Systems and Software", January 2009, vol. 82, n^o 1, pp. 3–22

[98] D. BENAVIDES, S. SEGURA, A. RUIZ-CORTES. *Automated Analysis of Feature Models 20 years Later: a Literature Review*, in "Information Systems", 2010, vol. 35, n^o 6

[99] N. BENCOMO. *On the use of software models during software execution*, in "MISE '09: Proceedings of the 2009 ICSE Workshop on Modeling in Software Engineering", IEEE Computer Society, May 2009

[100] T. BERGER, R. RUBLACK, D. NAIR, J. M. ATLEE, M. BECKER, K. CZARNECKI, A. WASOWSKI. *A survey of variability modeling in industrial practice*, in "VaMoS'13", ACM, 2013

[101] A. BEUGNARD, J.-M. JÉZÉQUEL, N. PLOUZEAU. *Contract Aware Components, 10 years after*, in "WCSI", 2010, pp. 1-11

[102] A. BLOUIN, B. COMBEMALE, B. BAUDRY, O. BEAUDOUX. *Kompren: Modeling and Generating Model Slicers*, in "Journal of Software and Systems Modeling (SoSyM)", 2012, pp. 1–17, http://dx.doi.org/10.1007/s10270-012-0300-x

[103] J. BOSCH. *Design and use of software architectures: adopting and evolving a product-line approach*, ACM Press/Addison-Wesley Publishing Co.New York, NY, USA, 2000

[104] J. BOSCH, G. FLORIJN, D. GREEFHORST, J. KUUSELA, J. H. OBBINK, K. POHL. *Variability Issues in Software Product Lines*, in "PFE '01: Revised Papers from the 4th International Workshop on Software Product-Family Engineering", London, UK, Springer-Verlag, 2002, pp. 13–21

[105] L. BRIAND, E. ARISHOLM, S. COUNSELL, F. HOUDEK, P. THÉVENOD–FOSSE. *Empirical studies of object-oriented artifacts, methods, and processes: state of the art and future directions*, in "Empirical Software Engineering", 1999, vol. 4, n^o 4, pp. 387–404

[106] J. T. BUCK, S. HA, E. A. LEE, D. G. MESSERSCHMITT. *Ptolemy: A framework for simulating and prototyping heterogeneous systems*, in "Int. Journal of Computer Simulation", 1994

[107] T. BURES, P. HNETYNKA, F. PLASIL. *Sofa 2.0: Balancing advanced features in a hierarchical component model*, in "Software Engineering Research, Management and Applications, 2006. Fourth International Conference on", IEEE, 2006, pp. 40–48

[108] G. BÉCAN, M. ACHER, B. BAUDRY, S. BEN NASR. *Breathing Ontological Knowledge Into Feature Model Management*, Inria, oct 2013, n° RT-0441, 15 p. , http://hal.inria.fr/hal-00874867

[109] B. H. C. CHENG, R. LEMOS, H. GIESE, P. INVERARDI, J. MAGEE, J. ANDERSSON, B. BECKER, N. BENCOMO, Y. BRUN, B. CUKIC, G. MARZO SERUGENDO, S. DUSTDAR, A. FINKELSTEIN, C. GACEK, K. GEIHS, V. GRASSI, G. KARSAI, H. M. KIENLE, J. KRAMER, M. LITOIU, S. MALEK, R. MIRANDOLA, H. A. MÜLLER, S. PARK, M. SHAW, M. TICHY, M. TIVOLI, D. WEYNS, J. WHITTLE. , D. HUTCHISON, T. KANADE, J. KITTLER, J. M. KLEINBERG, F. MATTERN, J. C. MITCHELL, M. NAOR, O. NIERSTRASZ, C. PANDU RANGAN, B. STEFFEN, M. SUDAN, D. TERZOPOULOS, D. TYGAR, M. Y. VARDI, G. WEIKUM, B. H. C. CHENG, R. LEMOS, H. GIESE, P. INVERARDI, J. MAGEE (editors) *Software Engineering for Self-Adaptive Systems: A Research Roadmap* , Betty H. C. Cheng, Rogério de Lemos, Holger Giese, Paola Inverardi, and Jeff Magee, Springer Berlin HeidelbergBerlin, Heidelberg, 2009, vol. 5525

[110] J. COPLIEN, D. HOFFMAN, D. WEISS. *Commonality and Variability in Software Engineering*, in "IEEE Software", 1998, vol. 15, n° 6, pp. 37–45

[111] I. CRNKOVIC, S. SENTILLES, A. VULGARAKIS, M. R. CHAUDRON. *A classification framework for software component models*, in "Software Engineering, IEEE Transactions on", 2011, vol. 37, n° 5, pp. 593–615

[112] K. CZARNECKI, U. W. EISENECKER. *Generative programming: methods, tools, and applications*, ACM Press/Addison-Wesley Publishing Co.New York, NY, USA, 2000

[113] R. DEMILLI, A. J. OFFUTT. *Constraint-based automatic test data generation*, in "Software Engineering, IEEE Transactions on", 1991, vol. 17, n° 9, pp. 900–910

[114] K. DEB, A. PRATAP, S. AGARWAL, T. MEYARIVAN. *A fast and elitist multiobjective genetic algorithm: NSGA-II*, in "Evolutionary Computation, IEEE Transactions on", 2002, vol. 6, n° 2, pp. 182–197

[115] S. FORREST, A. SOMAYAJI, D. H. ACKLEY. *Building diverse computer systems*, in "Operating Systems, 1997., The Sixth Workshop on Hot Topics in", IEEE, 1997, pp. 67–72

[116] R. FRANCE, B. RUMPE. *Model-driven Development of Complex Software: A Research Roadmap*, in "Proceedings of the Future of Software Engineering Symposium (FOSE '07)", L. C. BRIAND, A. L. WOLF (editors), IEEE, 2007, pp. 37–54

[117] S. FREY, F. FITTKAU, W. HASSELBRING. *Search-based genetic optimization for deployment and reconfiguration of software in the cloud*, in "Proceedings of the 2013 International Conference on Software Engineering", IEEE Press, 2013, pp. 512–521

[118] G. HALMANS, K. POHL. *Communicating the Variability of a Software-Product Family to Customers*, in "Software and System Modeling", 2003, vol. 2, n° 1, pp. 15-36

[119] C. HARDEBOLLE, F. BOULANGER. *ModHel'X: A component-oriented approach to multi-formalism modeling*, in "Models in Software Engineering", Springer, 2008, pp. 247–258

[120] M. HARMAN, B. F. JONES. *Search-based software engineering*, in "Information and Software Technology", 2001, vol. 43, n^{o} 14, pp. 833–839

[121] H. HEMMATI, L. C. BRIAND, A. ARCURI, S. ALI. *An enhanced test case selection approach for model-based testing: an industrial case study*, in "SIGSOFT FSE", 2010, pp. 267-276

[122] A. HUBAUX, M. ACHER, T. T. TUN, P. HEYMANS, P. COLLET, P. LAHIRE. *Separating Concerns in Feature Models: Retrospective and Support for Multi-Views*, in "Domain Engineering: Product Lines, Conceptual Models, and Languages", I. REINHARTZ-BERGER, A. STURM, T. CLARK, J. BETTIN, S. COHEN (editors), Springer, August 2013, pp. 3-28 [*DOI :* 10.1007/978-3-642-36654-3_1], http://hal.inria.fr/hal-00767213

[123] J. HUTCHINSON, J. WHITTLE, M. ROUNCEFIELD, S. KRISTOFFERSEN. *Empirical assessment of MDE in industry*, in "Proceedings of the 33rd International Conference on Software Engineering (ICSE '11)", R. N. TAYLOR, H. GALL, N. MEDVIDOVIC (editors), ACM, 2011, pp. 471–480

[124] J.-M. JÉZÉQUEL. *Model Driven Design and Aspect Weaving*, in "Journal of Software and Systems Modeling (SoSyM)", may 2008, vol. 7, n^{o} 2, pp. 209–218, http://www.irisa.fr/triskell/publis/2008/Jezequel08a.pdf

[125] K. C. KANG, S. G. COHEN, J. A. HESS, W. E. NOVAK, A. S. PETERSON. *Feature-Oriented Domain Analysis (FODA) Feasibility Study*, Carnegie-Mellon University Software Engineering Institute, November 1990

[126] J. KRAMER, J. MAGEE. *Self-Managed Systems: an Architectural Challenge*, in "Future of Software Engineering", IEEE, 2007, pp. 259–268

[127] K.-K. LAU, P. V. ELIZONDO, Z. WANG. *Exogenous connectors for software components*, in "Component-Based Software Engineering", Springer, 2005, pp. 90–106

[128] P. MCMINN. *Search-based software test data generation: a survey*, in "Software Testing, Verification and Reliability", 2004, vol. 14, n^{o} 2, pp. 105–156

[129] J. MEEKEL, T. B. HORTON, C. MELLONE. *Architecting for Domain Variability*, in "ESPRIT ARES Workshop", 1998, pp. 205-213

[130] A. M. MEMON. *An event-flow model of GUI-based applications for testing*, in "Software Testing, Verification and Reliability", 2007, vol. 17, n^{o} 3, pp. 137–157

[131] B. MORIN, O. BARAIS, J.-M. JÉZÉQUEL, F. FLEUREY, A. SOLBERG. *Models at Runtime to Support Dynamic Adaptation*, in "IEEE Computer", October 2009, pp. 46-53, http://www.irisa.fr/triskell/publis/2009/Morin09f.pdf

[132] B. MORIN, O. BARAIS, G. NAIN, J.-M. JÉZÉQUEL. *Taming Dynamically Adaptive Systems with Models and Aspects*, in "31st International Conference on Software Engineering (ICSE'09)", Vancouver, Canada, Canada, 2009, http://hal.inria.fr/inria-00468516

[133] P.-A. MULLER, F. FLEUREY, J.-M. JÉZÉQUEL. *Weaving Executability into Object-Oriented Meta-Languages*, in "Proc. of MODELS/UML'2005", Jamaica, LNCS, Springer, 2005

[134] R. MÉLISSON, P. MERLE, D. ROMERO, R. ROUVOY, L. SEINTURIER. *Reconfigurable run-time support for distributed service component architectures*, in "the IEEE/ACM international conference", New York, New York, USA, ACM Press, 2010, 171 p.

[135] G. NAIN. *EnTiMid : Un modèle de composants pour intégrer des objets communicants dans des applications à base de services*, Université Rennes 1, December 2011, http://hal.inria.fr/tel-00646664

[136] C. NEBUT, Y. LE TRAON, J.-M. JÉZÉQUEL. *System Testing of Product Families: from Requirements to Test Cases*, Springer Verlag, 2006, pp. 447–478, http://www.irisa.fr/triskell/publis/2006/Nebut06b.pdf

[137] C. NEBUT, S. PICKIN, Y. LE TRAON, J.-M. JÉZÉQUEL. *Automated Requirements-based Generation of Test Cases for Product Families*, in "Proc. of the 18th IEEE International Conference on Automated Software Engineering (ASE'03)", 2003, http://www.irisa.fr/triskell/publis/2003/nebut03b.pdf

[138] L. M. NORTHROP. *SEI's Software Product Line Tenets*, in "IEEE Softw.", 2002, vol. 19, n^{o} 4, pp. 32–40

[139] L. NORTHROP. *A Framework for Software Product Line Practice*, in "Proceedings of the Workshop on Object-Oriented Technology", Springer-Verlag London, UK, 1999, pp. 365–376

[140] I. OBER, S. GRAF, I. OBER. *Validating timed UML models by simulation and verification*, in "International Journal on Software Tools for Technology Transfer", 2006, vol. 8, n^{o} 2, pp. 128–145

[141] D. L. PARNAS. *On the Design and Development of Program Families*, in "IEEE Trans. Softw. Eng.", 1976, vol. 2, n^{o} 1, pp. 1–9

[142] S. PICKIN, C. JARD, T. JÉRON, J.-M. JÉZÉQUEL, Y. LE TRAON. *Test Synthesis from UML Models of Distributed Software*, in "IEEE Transactions on Software Engineering", April 2007, vol. 33, n^{o} 4, pp. 252–268

[143] K. POHL, G. BÖCKLE, F. J. VAN DER LINDEN. *Software Product Line Engineering: Foundations, Principles and Techniques*, Springer-Verlag New York, Inc.Secaucus, NJ, USA, 2005

[144] B. RANDELL. *System structure for software fault tolerance*, in "Software Engineering, IEEE Transactions on", 1975, n^{o} 2, pp. 220–232

[145] M. C. RINARD. *Obtaining and reasoning about good enough software*, in "Proceedings of Annual Design Automation Conference (DAC)", 2012, pp. 930-935

[146] J. ROTHENBERG, L. E. WIDMAN, K. A. LOPARO, N. R. NIELSEN. *The Nature of Modeling*, in "in Artificial Intelligence, Simulation and Modeling", John Wiley & Sons, 1989, pp. 75–92

[147] P. RUNESON, M. HÖST. *Guidelines for conducting and reporting case study research in software engineering*, in "Empirical Software Engineering", 2009, vol. 14, n^{o} 2, pp. 131–164

[148] D. SCHMIDT. *Guest Editor's Introduction: Model-Driven Engineering*, in "IEEE Computer", 2006, vol. 39, n^{o} 2, pp. 25–31

[149] F. SHULL, J. SINGER, D. I. SJBERG. *Guide to advanced empirical software engineering*, Springer, 2008

[150] S. SIDIROGLOU-DOUSKOS, S. MISAILOVIC, H. HOFFMANN, M. RINARD. *Managing performance vs. accuracy trade-offs with loop perforation*, in "Proc. of the Symp. on Foundations of software engineering", New York, NY, USA, ESEC/FSE '11, ACM, 2011, pp. 124-134

[151] J. STEEL, J.-M. JÉZÉQUEL. *On Model Typing*, in "Journal of Software and Systems Modeling (SoSyM)", December 2007, vol. 6, n° 4, pp. 401–414, http://www.irisa.fr/triskell/publis/2007/Steel07a.pdf

[152] C. SZYPERSKI, D. GRUNTZ, S. MURER. *Component software: beyond object-oriented programming*, Addison-Wesley, 2002

[153] J.-C. TRIGAUX, P. HEYMANS. *Modelling variability requirements in Software Product Lines: a comparative survey*, FUNDP Namur, 2003

[154] M. UTTING, B. LEGEARD. *Practical model-based testing: a tools approach*, Morgan Kaufmann, 2010

[155] P. VROMANT, D. WEYNS, S. MALEK, J. ANDERSSON. *On interacting control loops in self-adaptive systems*, in "SEAMS 2011", ACM, 2011, pp. 202–207

[156] C. YILMAZ, M. B. COHEN, A. A. PORTER. *Covering arrays for efficient fault characterization in complex configuration spaces*, in "Software Engineering, IEEE Transactions on", 2006, vol. 32, n° 1, pp. 20–34

[157] Z. A. ZHU, S. MISAILOVIC, J. A. KELNER, M. C. RINARD. *Randomized accuracy-aware program transformations for efficient approximate computations*, in "Proc. of the Symp. on Principles of Programming Languages (POPL)", 2012, pp. 441-454

[158] T. ZIADI, J.-M. JÉZÉQUEL. *Product Line Engineering with the UML: Deriving Products*, Springer Verlag, 2006, pp. 557-586