IN PARTNERSHIP WITH:
Institut national des sciences appliquées de Rennes

Université Rennes 1

Activity Report 2015

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

Creation of the Team: 2014 January 01, updated into Project-Team: 2014 July 01

Keywords:

Computer Science and Digital Science:

1.2.1. - Dynamic reconfiguration
2.1.10. - Domain-specific languages
2.1.2. - Object-oriented programming
2.5. - Software engineering
2.6.2. - Middleware
4.8. - Privacy-enhancing technologies

Other Research Topics and Application Domains:

3.1.1. - Resource management
6.1. - Software industry
6.4. - Internet of things
6.5. - Information systems
8.1.2. - Sensor networks for smart buildings

1. Members

Research Scientists

Benoit Baudry [Team leader, Inria, Researcher, HdR]
Benoit Combemale [Inria, Researcher, HdR]
Robert France [Colorado State University, PR, Advanced Research position, International Chaire, until Feb 2015]

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]
Noel Plouzeau [Univ. Rennes I, Associate Professor]

Engineers

Simon Allier [Inria]
Fabien Coulon [Inria]
Andre Elie [Inria]
Joao Bosco Ferreira Filho [Univ. Rennes I, until Jul 2015]
Jose Angel Galindo Duarte [Inria]
Aymeric Hervieu [Univ. Rennes I, until Feb 2015]
Manuel Leduc [Inria, from Aug 2015]
Dorian Leroy [Inria]
Jean Parpaillon [Inria]
Francois Tanguy [Inria, until Mar 2015]
Maxime Tricoire [Inria]

PhD Students

Francisco Javier Acosta Padilla [Univ. Rennes I]
Guillaume Becan [Univ. Rennes I]
Sana Ben Nasr [Inria]
Jacky Bourgeois [The Open University,UK]
Mohamed Boussaa [Inria]
Erwan Bousse [Inria]
Kevin Corre [Orange Labs, granted by CIFRE]
Thomas Degueule [Inria]
Inti Gonzalez Herrera [Univ. Rennes I, until Nov 2015]
Pierre Laperdrix [Univ. Rennes I]
Gwendal Le Moulec [INSA Rennes, from Oct 2015]
Valeria Lelli Leitao Dantas [Inria, until Oct 2015]
David Mendez Acuna [Inria]
Ivan Paez Anaya [Univ. Rennes I, until Apr 2015]
Johan Pelay [Inst. de Recherche Technologique B-COM]
Quentin Plazar [Inria, from Mar 2015]
Marcelino Rodriguez Cancio [Univ. Rennes I]
Paul Temple [Univ. Rennes I]
Kwaku Yeboah-Antwi [Inria]

Visiting Scientists

Jorg Kienzle [Mc Gill University, Montréal, PR, from Sep 2015]
Jorge Luis Rodas Silva [University of Milagro, PR, from Apr 2015]

Administrative Assistants

Tifenn Donguy [CNRS, from Oct 2015]
Loic Lesage [Inria, until Sep 2015]

Others

Francois Boschet [Inria, intern, from Jul 2015 until Aug 2015]
Simon Chopin [Inria, intern, from Jul 2015 until Aug 2015]
Francois Esnault [CNRS, intern, from May 2015 until Aug 2015]
Laureline Guerin [Inria, intern, from May 2015]
Oscar Guerin [Univ. Rennes I, intern, Jun 2015]
Stephane Mangin [Inria, intern, from May 2015 until Aug 2015]
Alexandre Rio [Univ. Rennes I, intern, from Jun 2015 until Sep 2015]
Owen Rouille [Normale Sup Rennes, intern, from May 2015 until Jul 2015]
Hugo Vallee [Inria, intern, from May 2015 until Aug 2015]
Olivier Beaudoux [OSEO, PR, Associate Member, until Aug 2015]
Gurvan Le Guernic [Research engineer, DGA, Associate member]

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) [132]. 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 [102]. 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 [81]. 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 [135].

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

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. [108] 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 [116].

3.1.2. Variability modeling

While the basic vision underlying *Software Product Lines* (SPL) can probably be traced back to David Parnas seminal article [125] 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 [122]. 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* [121]. Bosch provides a different definition [87]: *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 [96]. 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 [104]. 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 [88]. Variability management is thus growingly seen as the cornerstone of SPL development, covering the entire

development life cycle, from requirements elicitation [137] to product derivation [142] to product testing [120], [119].

Halmans *et al.* [104] 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.* [127]: 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.* [114] (feature, hardware platform, performances and attributes variability) or Bass *et al.* [79] 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* [110]. 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 [88]. 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 [83], [98]. Model Driven Engineering (MDE) makes it possible to automate this weaving process [109]. 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 [117]) manipulating these models to produce for instance a detailed design that can ultimately be transformed to code, or to test suites [126], or other software artifacts.

3.1.3. Component-based software development

Component-based software development [136] 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 [97]: description of requirements on services required from the other components; indirect connections between components thanks to ports and connectors constructs [112]; hierarchical definition of components (assemblies of components can define new component types); connectors supporting various communication semantics [91]; quantitative properties on the services [86].

In recent years component-based architectures have evolved from static designs to dynamic, adaptive designs (e.g., SOFA [91], Palladio [84], Frascati [118]). 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 [111]. 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 [95], [139]. 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 [85], 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 [130]". 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 [106] 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 [103]. Multi Objectives Search based techniques [100] 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 [138] mainly relies on model analysis, constraint solving [99] and search-based reasoning [113]. DIVERSE leverages in particular the applications

of model-based testing in the context of highly-configurable systems and [140] interactive systems [115] as well as recent advances based on diversity for test cases selection [107].

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 [123], 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 [90] 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 [80] provides modeling elements amenable to semantically equivalent mathematical models. Metropolis offers a precise semantics flexible enough to support different models of computation. ModHel'X [105] 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 [78], recovery blocks [128] and code randomization [82], 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 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 [133], [131]. Such methods have been used for example to understand the impact of new software development paradigms [89]. 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 [77].

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.

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 [132], [102]. 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.

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

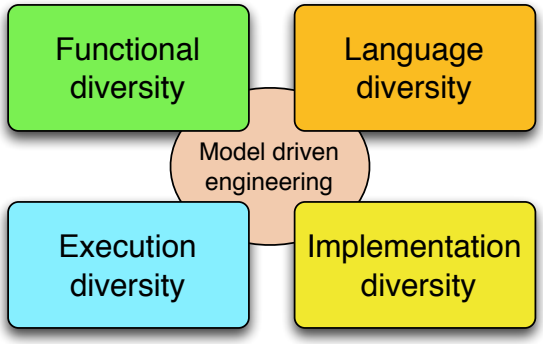

Figure 1. The four research axis of DIVERSE, which rely on a MDE scientific background

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 [134] (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 [128] and security [101] 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" [129], [141].

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. Highlights of the Year

5.1. Highlights of the Year

"Multi-tier diversification in Web-based software applications" appears in IEEE Software Magazine. This paper emphasizes a new type of software monoculture in Internet applications and introduces the idea of diversification in space and time at multiple levels of the software stacks. We experiment with a realistic Internet application to demonstrate the feasibility of multi-tier diversification. This experiment highlights the challenges that are ahead of software engineers if they want to systematically break the applicative monoculture of Internet applications.

The book "Globalizing Domain-Specific Languages" appears in the LNCS series. This book, edited by Benoit Combemale, Betty H.C. Cheng, Robert B. France, Jean-Marc Jézéquel, Bernhard Rumpe is the result of the Dagstuhl seminar organized by the GEMOC initiative in October 2014.

5.1.1. Awards

Ten years most influential paper award at MODELS'15 for the pioneering paper about the Kermeta meta-language [3] [117]

P.-A. Muller, F. Fleurey, J.-M. Jézéquel
Weaving executability into object-oriented meta-languages
Proc of MODELS/UML, p. 264-278, 2005.

6. New Software and Platforms

6.1. FAMILIAR

FeAture Model scrIpt Language for manIpulation and Automatic Reasoning

[3] http://www.cnrs.fr/ins2i/spip.php?article1733

KEYWORDS: Software line product - Configators - Customisation
SCIENTIFIC DESCRIPTION

FAMILIAR (for FeAture Model scrIpt Language for manIpulation and Automatic Reasoning) is a language for importing, exporting, composing, decomposing, editing, configuring, computing "diffs", refactoring, reverse engineering, testing, and reasoning about (multiple) feature models. All these operations can be combined to realize complex variability management tasks. A comprehensive environment is proposed as well as integration facilities with the Java ecosystem.

- Participants: Mathieu Acher, Guillaume Bécan, Sana Ben Nasr, Jose Galindo, Olivier Barais
- Contact: Mathieu Acher
- URL: http://familiar-project.github.com

6.2. OpenCompare

OpenCompare.org
KEYWORDS: product comparison matrix - tabular data - comparison
SCIENTIFIC DESCRIPTION

Product comparison matrices (PCMs) are tabular data: supported and unsupported features are documented for both describing the product itself and for discriminating one product compared to another. PCMs abound and constitute a rich source of knowledge for easily comparing and choosing product. Yet the current practice is suboptimal both for humans and computers, mainly due to unclear semantics, heterogeneous forms of data, and lack of dedicated support.

OpenCompare.org is a project for the collaborative edition, the sharing, the standardisation, and the open exploitation of PCMs. The goal of OpenCompare.org is to provide an integrated set of tools (e.g., APIs, visualizations, configurators, editors) for democratizing their creation, import, maintenance, and exploitation.

MatrixMiner is also part of opencompare. It is a tool for automatically synthesizing PCMs from a set of product descriptions written in natural language. MatrixMiner is capable of identifying and organizing features and values in a PCM despite the informality and absence of structure in the textual descriptions of products. More information here: https://matrix-miner.variability.io/

- Participants: Guillaume Bécan, Mathieu Acher, Sana Ben Nasr
- Contact: Mathieu Acher
- URL: http://opencompare.org

6.3. Kermeta

KEYWORDS: Eclipse - Model-driven engineering
SCIENTIFIC DESCRIPTION

Kermeta is used in several 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.
FUNCTIONAL DESCRIPTION

The Kermeta workbench is a powerful meta-programming environment based on an object-oriented DSL (Domain Specific Language) optimized for metamodel engineering. Kermeta is now integrated into Melange (see next tool), and will continue to be supported and developed under this new project with a broader scope.

- Participants: Zoé Drey, Cyril Faucher, Franck Fleurey, Jean Marc Jezequel, Pierre Alain Muller, Jim Steel, François Tanguy, Didier Vojtisek, Benoît Combemale, Olivier Barais, Arnaud Blouin, Benoit Baudry, Thomas Degueule, David Mendez Acuna, Erwan Bousse and Fabien Coulon
- Partners: Université de Rennes 1 - UHA
- Contact: Benoît Combemale
- URL: http://www.kermeta.org

6.4. Melange

KEYWORDS: Language workbench, Domain-Specific (Modeling) Language, Model-Driven Engineering, Model execution and debugging, Execution trace management
SCIENTIFIC DESCRIPTION

Melange is a follow-up of the executable metamodeling language Kermeta, which provides a tool-supported dedicated meta-language to safely assemble language modules, customize them and produce new DSMLs. Melange provides specific constructs to assemble together various abstract syntax and operational semantics artifacts into a DSML. DSMLs can then be used as first class entities to be reused, extended, restricted or adapted into other DSMLs. Melange relies on a particular model type system that statically ensures the structural correctness of the produced DSMLs, and specific subtyping relationships between DSMLs to reason about their substitutability. Newly produced DSMLs are correct by construction, ready for production (i.e., the result can be deployed and used as-is), and reusable in a new assembly.

Melange is a language workbench that support a modular and reusable approach for domain-specific language design and implementation.

FUNCTIONAL DESCRIPTION

Melange is a language workbench which helps language engineers to mashup their various language concerns as language design choices, to manage their variability, and support their reuse. It provides a modular and reusable approach for customizing, assembling and integrating DSMLs specifications and implementations. The language workbench embeds a model-oriented type system that provides model polymorphism and language substitutability, i.e. the possibility to manipulate a model through different interfaces and to define generic transformations that can be invoked on models written using different DSMLs. Melange also provides a dedicated meta-language where models are first-class citizens and languages are used to instantiate and manipulate them. By analogy with the class-based, object-oriented paradigm, Melange can be classified as a language-based, model-oriented programming language. Melange is tightly integrated with the Eclipse Modeling Framework ecosystem and relies on the meta-language Ecore for the definition of the abstract syntax of DSMLs. Executable meta-modeling is supported by weaving operational semantics defined with Kermeta (defined on top of Xtend). Melange is bundled as a set of Eclipse plug-ins.

- Participants: Thomas Degueule, Erwan Bousse, Fabien Coulon, Dorian Leroy, Didier Vojtisek, Olivier Barais, Arnaud Blouin, Benoit Combemale, Jean-Marc Jézéquel
- Partners: Université de Rennes 1
- Contact: Benoît Combemale
- URL: http://melange-lang.org

6.5. Kevoree

Kevoree Core C++
KEYWORDS: Embedded - Software Components - Software component - Dynamic adaptation
SCIENTIFIC DESCRIPTION

Kevoree is an open-source models@runtime platform (http://www.kevoree.org) 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.

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 remoteComponents 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 (http://frascati.ow2.org).
- SpringSource Dynamic Module (http://spring.io/)
- GCM-Proactive (http://proactive.inria.fr/)
- OSGi (http://www.osgi.org)
- Chef
- Vagran (http://vagrantup.com/)

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:

Several tutorials and courses have been performed this year at EJCP for French PhD student, at ECNU summer school for 82 chineese PhD students. See also the web page http://www.kevoree.org .

In 2015, we mainly created a new implementation in C# 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: 5.3.1

Programming language: Java, Scala, Kermeta, Kotlin, Javascript, c#

FUNCTIONAL DESCRIPTION

Kevoree is an open-source models@runtime platform to properly support the dynamic adaptation of distributed systems. Models@runtime basically pushes the idea of reflection 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.

- Participants: Olivier Barais, Johann Bourcier, Noël Plouzeau, Benoit Baudry, Maxime Tricoire, Jacky Bourgeois, Inti Gonzalez Herrera, Ivan Paez Anaya, Manuel Leduc, Francisco-Javier Acosta Padilla and Mohamed Boussaa
- Partner: Université de Rennes 1
- Contact: Olivier Barais
- URL: http://kevoree.org/

6.6. amiunique

KEYWORDS: Privacy - Browser fingerprinting
FUNCTIONAL DESCRIPTION

This web site aims at informing visitors about browser fingerprinting and possible tools to mitigate its effect, as well as at collecting data about the fingerprints that can be found on the web. It collects browser fingerprints with the explicit agreement of the users (they have to click on a button on the home page). Fingerprints are composed of 17 attributes, which include regular HTTP headers as well as the most recent state of the art techniques (canvas fingerprinting, WebGL information).

SCIENTIFIC DESCRIPTION

The amiunique web site has been deployed in the context of the DiverSE's research activities on browser fingerprinting and how software diversity can be leveraged in order to mitigate the impact of fingerprinting on the privacy of users. The construction of a dataset of genuine fingerprints is essential to understand in details how browser fingerprints can serve as unique identifiers and hence what should be modified in order to mitigate its impact privacy. This dataset also supports the large-scale investigation of the impact of web technology advances on fingerprinting. For example, we can analyze in details the impact of the HTML5 canvas element or the behavior of fingerprinting on mobile devices.

The whole source code of amiunique is open source and is distributed under the terms of the MIT license.

Similar sites:

- Panopticlick https://panopticlick.eff.org/
- BrowserSpy http://browserspy.dk/
- http://noc.to/

Main innovative features:

- canvas fingerprinting
- WebGL fingerprinting
- advanced JS features (platform, DNT, etc.)

Impact:

The website has been showcased in several professional forums in 2014 and 2015 (Open World Forum 2015, FOSSA'14, FIC'15, ICT'15) and it has been visited by more than 100000 unique visitors in one year.

Programming language: Java, JavaScript, Scala

- Participants: Pierre Laperdrix, Benoit Baudry
- Partner: INSA Rennes
- Contact: Benoit Baudry
- URL: https://amiunique.org/
- URL source code: https://github.com/DIVERSIFY-project/amiunique

7. New Results

7.1. Results on Software Language Engineering

7.1.1. Modular and Reusable Development of DSLs

Domain-Specific Languages (DSLs) are now developed for a wide variety of domains to address specific concerns in the development of complex systems. When engineering new DSLs, it is likely that previous efforts spent on the development of other languages could be leveraged, especially when their domains overlap. However, legacy DSLs may not fit exactly the end user requirements and thus require further extension, restriction, or specialization. While current language workbenches provide import mechanisms, they usually lack an explicit support for such customizations of imported artifacts. We propose an approach for building DSLs by safely assembling and customizing legacy DSLs artifacts. This approach is based on typing relations that provide a reasoning layer for manipulating DSLs while ensuring type safety. On top of this reasoning layer, we provide an algebra of operators for extending, restricting, and assembling separate DSL artifacts. We implemented the typing relations and algebra into the Melange meta-language [30], [29], [73].

7.1.2. Executable Domain-Specific Modeling Languages (xDSMLs)

Executable Domain-Specific Modeling Languages (xDSMLs) open many possibilities for performing early verification and validation (V&V) of systems. Dynamic V&V approaches rely on execution traces, which represent the evolution of models during their execution. In order to construct traces, generic trace metamodels can be used. Yet, regarding trace manipulations, they lack both efficiency because of their sequential structure, and usability because of their gap to the xDSML. We contributed a generative approach that defines a rich and domain-specific trace metamodel enabling the construction of execution traces for models conforming to a given xDSML [24]. We also contributed a partly generic omniscient debugger supported by generated domain-specific trace management facilities [49].

The emergence of modern concurrent systems calls for xDSMLs where concurrency is of paramount importance. Such xDSMLs are intended to propose constructs with rich concurrency semantics, which allow system designers to precisely define and analyze system behaviors. In [34], we introduce a concurrent executable metamodeling approach, which supports a modular definition of the execution semantics, including the concurrency model, the semantic rules, and a well-defined and expressive communication protocol between them. In [28], we present MoCCML, a dedicated meta-language for formally specifying the concurrency concern within the definition of a DSL. 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. MoCCML is illustrated on the definition of an lightweight extension of SDF (SynchronousData Flow).

7.1.3. Globalization of Domain-Specific Modeling Languages

The development of modern complex software-intensive systems often involves the use of multiple DSMLs that capture different system aspects. Supporting coordinated use of DSMLs leads to what we call the globalization of modeling languages, that is, the use of multiple modeling languages to support coordinated development of diverse aspects of a system.

In a book published in 2015 [66], a number of articles describe the vision and the way globalized DSMLs currently assist integrated DSML support teams working on systems that span many domains and concerns to determine how their work on a particular aspect influences work on other aspects. Globalized DSMLs offer support for communicating relevant information, and for coordinating development activities and associated technologies within and across teams, in addition to providing support for imposing control over development artifacts produced by multiple teams. DSMLs can be used to support socio-technical coordination by providing the means for stakeholders to bridge the gap between how they perceive a problem and its solution, and the programming technologies used to implement a solution. They also support coordination of work across

multiple teams. DSMLs developed in an independent manner to meet the specific needs of domain experts have an associated framework that regulates interactions needed to support collaboration and work coordination across different system domains. The book includes [63], [65], [64], [62] with authors from the DIVERSE team.

In [43], we propose a Behavioral Coordination Operator Language (B-COOL) to reify coordination patterns between specific domains by using coordination operators between the Domain-Specific Modeling Languages used in these domains. Those operators are then used to automate the coordination of models conforming to these languages. We illustrate the use of B-COOL with the definition of coordination operators between timed finite state machines and activity diagrams.

The GEMOC Studio (http://gemoc.org/studio) is an eclipse package that contains components for building and composing executable Domain-Specific Modeling Languages (DSMLs). The GEMOC Studio complements Melange to formally define in a modular way the concurrency model of executable DSMLs, and provides analysis and coordination facilities based on the concurrency model. It also integrates all the contributions presented in this document related to model execution, animation, debugging and trace management. The GEMOC studio has been the overall winner of the transformation tool contest 2015 on Model Execution [52].

7.1.4. An analysis of metamodeling practices for MOF and OCL

The definition of a metamodel that precisely captures domain knowledge for effective know-how capitalization is a challenging task. A major obstacle for domain experts who want to build a metamodel is that they must master two radically different languages: an object-oriented, MOF-compliant, modeling language to capture the domain structure and first order logic (the Object Constraint Language) for the definition of well-formedness rules. However, there are no guidelines to assist the conjunct usage of both paradigms, and few tools support it. Consequently, we observe that most metamodels have only an object-oriented domain structure, leading to inaccurate metamodels. In [21], we perform the first empirical study, which analyzes the current state of practice in metamodels that actually use logical expressions to constrain the structure. We analyze 33 metamodels including 995 rules coming from industry, academia and the Object Management Group, to understand how metamodelers articulate both languages. We implement a set of metrics in the OCLMetrics tool to evaluate the complexity of both parts, as well as the coupling between both. We observe that all metamodels tend to have a small, core subset of concepts, which are constrained by most of the rules, in general the rules are loosely coupled to the structure and we identify the set of OCL constructs actually used in rules.

7.1.5. Model Slicers

Among model comprehension tools, model slicers are tools that extract a subset of model elements, for a specific purpose. We propose the Kompren language to model and generate model slicers for any DSL (*e.g.* modeling for software development or for civil engineering) and for different purposes (*e.g.* monitoring and model comprehension). We detail the semantics of the Kompren language and of the model slicer generator. This provides a set of expected properties about the slices that are extracted by the different forms of the slicer [18]. We show how the use of Kompren, a domain-specific language for defining model slicers, can ease the development of such interactive visualization features [19].

In Model Driven Development (MDD), it is important to ensure that a model conforms to the invariants defined in the metamodel. General-purpose rigorous analysis tools that check invariants are likely to perform the analysis over the entire metamodel and model. Since modern day software is exceedingly complex, the size of the model together with the metamodel can be very large. Consequently, invariant checking can take a very long time.To this end, we introduce model slicing within the invariant checking process, and use a slicing technique to reduce the size of the inputs in order to make invariant checking of large models feasible with existing tools [22], [42].

7.1.6. Bridging the gap between scientific models and engineering models with MDE

The complex problems that computational science addresses are more and more benefiting from the progress of computing facilities (e.g., simulators, librairies, accessible languages). Nevertheless, the actual solutions

call for several improvements. Among those, we address in [25] the needs for leveraging on knowledge and expertise by focusing on Domain-Specific Modeling Languages application. In this vision paper we illustrate, through concrete experiments, how the last DSML research help getting closer the problem and implementation spaces.

Various disciplines use models for different purposes. While engineering models, including software engineering models, are often developed to guide the construction of a non- existent system, scientific models, in contrast, are created to better understand a natural phenomenon (i.e., an already existing system). An engineering model may incorporate scientific models to build a system. Both engineering and scientific models have been used to support sustainability, but largely in a loosely-coupled fashion, independently developed and maintained from each other. Due to the inherent complex nature of sustainability that must balance trade-offs between social, environmental, and economic concerns, modeling challenges abound for both the scientific and engineering disciplines. In [72] we propose a vision that synergistically combines engineering and scientific models to enable broader engagement of society for addressing sustainability concerns, informed decision-making based on more accessible scientific models and data, and automated feed-back to the engineering models to support dynamic adaptation of sustainability systems. To support this vision, we identify a number of challenges to be addressed with particular emphasis on the socio-technical benefits of modeling.

As first experiments, we presented at the Inria-Industry meeting 2015 on energy transition and EclipseCon 2015, an approach to develop smart cyber physical systems in charge of managing the production, distribution and consumption of energies (e.g., water, electricity). The main objective is to enable a broader engagement of society, while supporting a more informed decision-making, possibly automatically, on the development and run-time adaptation of sustainability systems (e.g., smart grid, home automation, smart cities). We illustrate this approach through a system that allows farmers to simulate and optimize their water consumption by combining the model of a farming system together with agronomical models (e.g., vegetable and animal lifecycle) and open data (e.g., climate series). To do so, we use Model Driven Engineering (MDE) and Domain Specific Languages (DSL) to develop such systems driven by scientific models that define the context (e.g., environment, social and economy), and model experiencing environments to engage general public and policy makers.

7.2. Results on Variability Modeling and Engineering

7.2.1. Reverse engineering variability

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 [93]. We have performed an empirical evaluation on hundreds of feature models, coming from the SPLOT repository and Wikipedia [92]. 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 [20]). We have also considered numerical information and feature *attributes* so that we are now capable of synthesizing attributed feature models from product descriptions [51].

Besides, we have developed techniques for reverse engineering variability in generators and configurators (e.g., video generators) [50]. We have identified new research directions for protecting variability [44] mainly due to the fact reverse engineering techniques (previously presented) are effective .

7.2.2. 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. PCMs can be seen as a formalism for modeling a family of products, including variability information. Despite their apparent simplicity, PCMs contain heterogeneous, ambiguous, uncontrolled and partial information that hinders their efficient exploitations. We have formalized PCMs through model-based automated techniques and developed additional tooling to support the edition and

re-engineering of PCMs [94]. 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 https://opencompare.org/

Another axis is the mining of PCMs since (1) the manual elaboration of PCMs has limitations (2) numerous sources of information can be combined and are amenable to PCMs. We have developed MatrixMiner a tool for automatically synthesizing PCMs from a set of product descriptions written in natural language [46]. MatrixMiner is capable of identifying and organizing features and values in a PCM despite the informality and absence of structure in the textual descriptions of products. More information is available online: https://matrix-miner.variability.io/

7.3. Results on Heterogeneous and dynamic software architectures

7.3.1. Resource Monitoring and Reservation in Heterogeneous and dynamic software architectures

Software systems are more pervasive than ever nowadays. Occasionally, applications run on top of resource-constrained devices where efficient resource management is required; hence, they must be capable of coping with such limitations. However, applications require support from the runtime environment to properly deal with resource limitations. This thesis addresses the problem of supporting resource-aware programming in execution environments. In particular, it aims at offering efficient support for collecting data about the consumption of computational resources (e.g., CPU, memory), as well as efficient mechanisms to reserve resources for specific applications. In existing solutions we find two important drawbacks. First, they impose performance overhead on the execution of applications. Second, creating resource management tools for these abstractions is still a daunting task. The outcomes of this work [12] are three contributions:

- An optimistic resource monitoring framework that reduces the cost of collecting resource consumption data.
- A methodology to select components' bindings at deployment time in order to perform resource reservation.
- A language to build customized memory profilers that can be used both during applications' development, and also in a production environment.

7.3.2. Dynamic Reasoning on Heterogeneous and dynamic software architectures

Multi-Objective Evolutionary Algorithms (MOEAs) have been successfully used to optimize various domains such as finance, science, engineering, logistics and software engineering. Nevertheless, MOEAs are still very complex to apply and require detailed knowledge about problem encoding and mutation operators to obtain an effective implementation. Software engineering paradigms such as domain-driven design aim to tackle this complexity by allowing domain experts to focus on domain logic over technical details. Similarly, in order to handle MOEA complexity, we propose an approach, using model-driven software engineering (MDE) techniques, to define fitness functions and mutation operators without MOEA encoding knowledge. Integrated into an open source modelling framework, our approach can significantly simplify development and maintenance of multi-objective optimizations. By leveraging modeling methods, our approach allows reusable optimizations and seamlessly connects MOEA and MDE paradigms. We evaluate our approach on a cloud case study and show its suitability in terms of i) complexity to implement an MOO problem, ii) complexity to adapt (maintain) this implementation caused by changes in the domain model and/or optimization goals, and iii) show that the efficiency and effectiveness of our approach [56] remains comparable to ad-hoc implementations.

7.3.3. A Precise Metamodel for Open Cloud Computing Interface

Open Cloud Computing Interface (OCCI) proposes one of the first widely accepted, community-based, open standards for managing any kinds of cloud resources. But as it is specified in natural language, OCCI is imprecise, ambiguous, incomplete, and needs a precise definition of its core concepts. Indeed, the OCCI Core Model has conceptual drawbacks: an imprecise semantics of its type classification system, a nonextensible data type system for OCCI attributes, a vague and limited extension concept and the absence of a configuration concept. To tackle these issues, this work proposes a precise metamodel for OCCI. This metamodel defines rigourously the static semantics of the OCCI core concepts, of a precise type classification system, of an extensible data type system, and of both extension and configuration concepts. This metamodel is based on the Eclipse Modeling Framework (EMF), its structure is encoded with Ecore and its static semantics is rigourously defined with Object Constraint Language (OCL). As a consequence, this metamodel provides a concrete language to precisely define and exchange OCCI models. The validation of our metamodel is done on the first worldwide dataset of OCCI extensions already published in the literature, and addressing inter-cloud networking, infrastructure, platform, application, service management, cloud monitoring, and autonomic computing domains, respectively. This validation highlights simplicity, consistency, correctness, completeness, and usefulness of the proposed metamodel[38], [41].

7.3.4. Using Novelty Search Approach and models@runtime for Automatic Testing Environment Setup

In search-based structural testing, metaheuristic search techniques have been frequently used to automate the test data generation. In Genetic Algorithms (GAs) for example, test data are rewarded on the basis of an objective function that represents generally the number of statements or branches covered. However, owing to the wide diversity of possible test data values, it is hard to find the set of test data that can satisfy a specific coverage criterion. In this work, we introduce the use of Novelty Search (NS) algorithm to the test data generation problem based on statement-covered criteria. We believe that such approach to test data generation is attractive because it allows the exploration of the huge space of test data within the input domain. In this approach, we seek to explore the search space without regard to any objectives. In fact, instead of having a fitness-based selection, we select test cases based on a novelty score showing how different they are compared to all other solutions evaluated so far [47], [48]. We also create an architecture generation framework for setup testing environment for a distributed and heterogeneous service.

7.3.5. Using Models@Run.time to embed an Energetic Cloud Simulator in a MAPE-K Loop

Due to high electricity consumption in the Cloud datacenters, providers aim at maximizing energy efficiency through VM consolidation, accurate resource allocation or adjusting VM usage. More generally, the provider attempts to optimize resource utilization. However, while minimizing expenses, the Cloud operator still needs to conform to SLA constraints negotiated with customers (such as latency, downtime, affinity, placement, response time or duplication). Consequently, optimizing a Cloud configuration is a multi-objective problem. As a nontrivial multi-objective optimization problem, there does not exist a single solution that simultaneously optimizes each objective. There exists a (possibly infinite) number of Pareto optimal solutions. Evolutionary algorithms are popular approaches for generating Pareto optimal solutions to a multi-objective optimization problem. Most of these solutions use a fitness function to assess the quality of the candidates. However, regarding the energy consumption estimation, the fitness function can be approximative and lead to some imprecisions compared to the real observed data. This work presents a system that uses a genetic algorithm to optimize Cloud energy consumption and machine learning techniques to improve the fitness function regarding a real distributed cluster of server. We have carried out experiments on the OpenStack platform to validate our solution. This experimentation shows that the machine learning produces an accurate energy model, predicting precise values for the simulation [124][40].

7.4. 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 (i) new approaches to increase diversity in software systems, and (ii) software testing to assess the validity of software.

7.4.1. Software diversification

Early experiments with software diversity in the mid 1970's investigated N-version programming and recovery blocks to increase the reliability of embedded systems. Four decades later, the literature about software diversity has expanded in multiple directions: goals (fault-tolerance, security, software engineering); means (managed or automated diversity) and analytical studies (quantification of diversity and its impact). We contribute to the field of software diversity with the very first literature survey that adopts an inclusive vision of the area, with an emphasis on the most recent advances in the field. This survey includes classical work about design and data diversity for fault tolerance, as well as the cybersecurity literature that investigates randomization at different system levels. It broadens this standard scope of diversity, to include the study and exploitation of natural diversity and the management of diverse software products [17].

We also contribute to software diversity with novel techniques and methods. The interdisciplinary investigations within the DIVERSIFY project have led to the definition of novel principles for open-ended evolution in software systems. The main intuition is that software should have the ability to spontaneously and continuously evolve without waiting for specific environmental conditions. Our proposal analogizes the software consumer / provider network, which can be found in any types of distributed systems, to a bipartite ecological graph. This analogy provides the foundations for the design of an individual-based simulator used to experiment with decentralized adaptation strategies for providers and consumers. The initial model of a software network is tuned according to observations gathered from real-world software networks. The key insights about our experiments are that, 1) we can successfully model software systems as an ALife system, and 2) we succeed in emerging a global property from local decisions: when consumers and providers adapt with local decision strategies, the global robustness of the network increases. We show that these results hold with different initial situations, different scales and different topological constraints on the network [55]. In order to move towards the open-ended evolution of actual systems, we also developed a novel tool for the runtime modification of Java programs, as an extension to the JVM [60].

Our second contribution to the filed of software diversity consists in experimenting its application in different fields. First, we have proposed a novel approach to exploit software diversity at multiple granularity levels simultaneously [15]. The main idea is to reconcile two aspects of the massive software reuse in web applications: on the one hand, reuse and modularity favor much writing the next killer application; on the other hand, reuse and modularity facilitates much the next massive BOBE attack. We demonstrate the feasiblity of diversifying web applications at multiple levels, mitigating the risks of reuse.

The second application of automatic software diversification for Java programs aimed at answering the following question: which product line operators, applied to which program elements , can synthesize variants of programs that are incorrect , correct or perhaps even conforming to test suites? We implement source code transformations, based on the derivation operators of the Common Variability Language. We automatically synthesize more than 370,000 program variants from a set of 8 real large Java projects (up to 85,000 lines of code), obtaining an extensive panorama of the sanity of the operations [68].

The third application of software diversification is against browser fingerprinting. Browser fingerprint tracking relies on the following mechanisms: web browsers allow remote servers to discover sufficient information about a user's platform to create a digital fingerprint that uniquely identifies the platform. We argue that fingerprint uniqueness and stability are the key threats to browser fingerprint tracking, and we aim at breaking fingerprint stability over time, by exploiting software diversity and automatic reconfiguration. We leverage virtualization and modular software architectures to automatically assemble and reconfigure a user's software components at multiple levels. We operate on the operating system, the browser, the lists of fonts and plugins. This work is the first application of software reconfiguration to build a moving target defense against browser

fingerprint tracking. We have developed a prototype called *Blink* to experiment the effectiveness of our approach at randomizing fingerprints [33].

7.4.2. Software testing

Our work in the area of software testing focuses on tailoring the testing tools (analysis, generation, oracle, etc.) to specific domains. This allows us to consider domain specific knowledge (e.g., architectural patterns for GUI implementation) in order to increase the relevance and the efficiency of testing. The main results of this year are about testing GUIs and model transformations.

Graphical user interfaces (GUIs) are integral parts of software systems that require interactions from their users. Software testers have paid special attention to GUI testing in the last decade, and have devised techniques that are effective in finding several kinds of GUI errors. However, the introduction of new types of interactions in GUIs presents new kinds of errors that are not targeted by current testing techniques. We believe that to advance GUI testing, the community needs a comprehensive and high level GUI fault model, which incorporates all types of interactions. In this work, we first propose a GUI fault model designed to identify and classify GUI faults [37]. We then studied the impact of the new types of interactions in GUIs on their testing process. We show that the current GUI model-based testing approaches have limits when applied to test such new advanced GUIs [36].

Specifying a model transformation is challenging as it must be able to give a meaningful output for any input model in a possibly infinite modeling domain. Transformation preconditions constrain the input domain by rejecting input models that are not meant to be transformed by a model transformation. In our latest work [39], we present a systematic approach to discover such preconditions when it is hard for a human developer to foresee complex graphs of objects that are not meant to be transformed. The approach is based on systematically generating a finite number of test models using our tool, PRAMANA to first cover the input domain based on input domain partitioning. Tracing a transformation's execution reveals why some preconditions are missing. Using a benchmark transformation from simplified UML class diagram models to RDBMS models we discover new preconditions that were not initially specified.

We also initiated a new line of research in order to invesitgate Novelty Search (NS) for the automatic generation of test data. This allows the exploration of the huge space of test data within the input domain. In this approach, we select test cases based on a novelty score showing how different they are compared to all other solutions evaluated so far [47].

In Model Driven Engineering (MDE), models are first-class citizens, and model transformation is MDE's "heart and soul". Since model transformations are executed for a family of (conforming) models, their validity becomes a crucial issue. In [16] we proposes to explore the question of the formal verification of model transformation properties through a tridimensional approach: the transformation involved, the properties of interest addressed, and the formal verification techniques used to establish the properties. This work is intended for a double audience. For newcomers, it provides a tutorial introduction to the field of formal verification of model transformations. For readers more familiar with formal methods and model transformations, it proposes a literature review (although not systematic) of the contributions of the field. Overall, this work allows to better understand the evolution, trends and current practice in the domain of model transformation verification. This work opens an interesting research line for building an engineering of model transformation verification guided by the notion of model transformation intent.

8. Bilateral Contracts and Grants with Industry

8.1. Bilateral Contracts with Industry

8.1.1. April

This work is performed in collaboration with APRIL Technologies. This company develops all the IT solutions for APRIL group [4] and their clients in the insurance business. They have a very large information system that they specialize for all the divisions of the group. A critical need for them is to ensure that changes in their applications (new features, bug repair, etc.) do not degrade functional correctness and performance.

Software testing techniques and tools have greatly improved over the last decade and it is now possible for software developers to write test cases that are automatically executed. Consequently, each time the program evolves, it is rebuild and re-tested automatically, which supports the detection of errors early in the process and prevents the propagation of the bug into the production code. However, the test cases are manually written and are thus usually weak when it comes at finding bugs that are deep in the code or in nested loops for example. The main challenge of this work is automatically generate new test cases that increase the effectiveness of regression testing.

In this project we aim at automatically generating new test cases from the ones that have been manually produced by the developers, in order to add value in the continuous integration process and improve the quality of software that goes in production. The process of automatically producing new test cases from existing ones is called *test amplification*. We can experiment our recent results about test transformations on APRIL Technologies's set of test cases very early in the project.

This project supports one postdoc in the DiverSE team and is funded by Inria's transfer and industrial partnership department.

8.2. Bilateral Grants with Industry

8.2.1. 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.

9. Partnerships and Cooperations

9.1. National Initiatives

9.1.1. ANR

9.1.1.1. GEMOC

- Coordinator: Inria (DIVERSE)
- Other partners: ENSTA Bretagne, Inria, IRIT, I3S, Obeo, Thales
- Dates: 2012-2016

[4] http://groupe.april.fr/groupe

- 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.

9.1.1.2. 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.

9.1.1.3. 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.

9.1.2. BGLE / LEOC

9.1.2.1. 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.

9.1.2.2. 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).

9.1.2.3. 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.

9.1.3. DGA

9.1.3.1. 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.

9.1.3.2. 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.

9.2. European Initiatives

9.2.1. FP7 & H2020 Projects

9.2.1.1. FP7 FET STREP DIVERSIFY

- Coordinator: Inria (DIVERSE)
- Partners: SINTEF, Université de Rennes 1, Trinity College Dublin, Inria (DiverSE, SPIRALS)
- 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.

9.2.1.2. 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.

9.2.2. Collaborations in European Programs, except FP7 & H2020

9.2.2.1. 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. Benoit Combemale is a member of the management committee.

9.2.2.2. 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.

9.2.3. 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.

9.3. International Initiatives

9.3.1. Inria International Partners

9.3.1.1. Declared Inria International Partners

9.3.1.1.1. Inria International Chair

Prof. Robert B. France [5] 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.

9.3.1.2. Informal International Partners

- Université de Montréal (Canada)
- McGill University (Canada)
- University of Alabama (USA)
- TU Wien (Austria)
- Michigan State University (MSU)
- Aachen University (Germany)

9.3.2. 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.

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

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.

9.4. International Research Visitors

9.4.1. Visits of International Scientists

- Prof. Jörg Kienzle from McGill University (Canada) has been in the DIVERSE team during his Sabbatical from September 2015 to December 2015.
- Prof. Betty H.C. Cheng from Michigan State University (USA) visited the DIVERSE team in December 2015.
- Dr. Franck Fleurey from Sintef visited the DIVERSE team two weeks the team in July and November 2015.

10. Dissemination

10.1. Promoting Scientific Activities

10.1.1. Scientific events organisation

10.1.1.1. Member of the organizing committees

Benoit Baudry:

- International Symposium on Software Testing and Analysis (ISSTA'15)
- International Conference on Model Driven Engineering, Languages and Systems (MODELS'15)

Benoit Combemale:

- International Conference on Model Driven Engineering, Languages and Systems (MODELS'15)
- International Workshop on The Globalization of Modeling Languages (GEMOC'15)
- international workshop on Next Generation of Modularity Approaches for Multiple Dimensions of Sustainability (Sustainability'15)

10.1.2. Scientific events selection

10.1.2.1. Chair of conference program committees

Jean-Marc Jézéquel is co-chair of Digital Intelligence 2016, April 2016, Quebec City, Canada. Noel Plouzeau is co-chair of CBSE 2016 (Component Based Software Engineering) conference program committee, April 2016, Venice, Italy.

10.1.2.2. Member of the conference program committees

Benoit Baudry:

- International Conference on Software Engineering (ICSE'15)
- International Conference on Foundations of Software Engineering (ESEC-FSE'15)
- International Conference on Automated Software Engineering (ASE'15)
- International Conference on Model Driven Engineering, Languages and Systems (MODELS'15)

Arnaud Blouin:

- IEEE International Workshop on User Centered Design and Adaptive Systems (UCDAS), 2015

Benoit Combemale:

- International Conference on Software Language Engineering (SLE'15)
- European Conference on Modelling Foundations and Applications (ECMFA'15)
- International Conference on Model Transformation (ICMT'15)
- International Conference on Software Engineering and Formal Methods (SEFM'15)
- International Conference on Software and System Process (ICSSP'15)
- International Workshop on Multi-Paradigm Modeling (MPM'15)
- International Workshop on Executable Modeling (EXE'15)
- International Workshop on Modelling in Software Engineering (MiSE'2015)

Jean-Marc Jézéquel:

- 10th International Workshop on Models at Run-Time, associated to MODELS 2015
- 19th International Software Product Line Conference, July 20-24, Nashville, TN USA.
- 9th International Workshop on Variability Modelling of Software-intensive Systems, January 22-24, 2015 - Hildesheim, Germany

Johann Bourcier:

- 1st International Workshop on Software Engineering for Smart Cyber-Physical Systems, associated to ICSE 2015
- 3rd FoCAS Workshop on Fundamentals of Collective Adaptive Systems, associated to SASO 2015

Mathieu Acher:

- 19th International Software Product Line Conference, July 20-24, Nashville, TN USA.
- SEAA'15 : Euromicro Conference series on Software Engineering and Advanced Applications
- SAC'15 (SE) : 30th ACM Symposium on Applied Computing - Software Engineering (SE) Track
- ICSE'15 (NIER) : 37th International Conference on Software Engineering - New Ideas and Emerging Results (NIER) Track
- EduSymp'15 : Educators Symposium @ Model Driven Engineering Languages and Systems
- A-MOST'15 : 11th Workshop on Advances in Model Based Testing (at ICST)
- SPLTea'15: 2nd International workshop on teaching and product lines (organizer)

Noel Plouzeau:

- CBSE 2015: 18th conference on Component Based Software Engineering, Montreal, Canada, 2015.

Olivier Barais:

- Modularity 2015 Demo Poster Session: 14th International Conference on Modularity March 16-19, 2015 Ft. Collins, Colorado, USA
- SEAA 2015: 41st EUROMICRO Conference on Software Engineering and Advanced Applications (SEAA) Madeira, Portugal, August 26th - 28th, 2015
- SAC 2015: SAC 2015 The 30th ACM/SIGAPP Symposium On Applied Computing Salamanca, Spain
- SPLat 2015: 2nd Software Product Line Analysis Tools Workshop, July 20-24, Nashville, TN USA.
- WETICE 2016: 24th IEEE International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises.

10.1.2.3. Reviewer

Arnaud Blouin:

- EICS'15, ACM SIGCHI symposium on Engineering interactive computing systems
- ICSE'15, International Conference on Software Engineering
- IHM'15, French speaking conference on human-computer interaction

Johann Bourcier

- ICSE'15, 37th International Conference on Software Engineering
- FSE'15, 10th Joint Meeting of the European Software Engineering Conference and the ACM SIGSOFT Symposium on the Foundations of Software Engineering
- CBSE'15, The 18th International ACM Sigsoft Symposium on Component-Based Software Engineering
- DSN 2015, The 45th IEEE/IFIP International Conference on Dependable Systems and Networks

10.1.3. Journal

10.1.3.1. Member of the editorial boards

Benoit Baudry is an Associate Editor of the following journals:

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

Benoit Combemale is member of the editorial boards of the following journals:

- Journal on Computer Languages, Systems and Structures (COMLAN)
- Journal on Science of Computer Programming (SCP), Advisory Board of the Software Section.

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

- Journal on Software and System Modeling (SoSYM)

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

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

10.1.3.2. Reviewer - Reviewing activities

Arnaud Blouin:

- Journal Universal Access in the Information Society
- Journal of Computer Science and Technology
- Journal of Software Engineering for Robotics

Benoit Combemale:

- Journal on Software and System Modeling (SoSYM)
- Transactions on Software Engineering (TSE)

Mathieu Acher

- Journal on Software and System Modeling (SoSYM)
- Transactions on Software Engineering (TSE)
- Information and Software Technology (IST)
- Automated Software Engineering: An International Journal (ASE)

Johann Bourcier:

- Journal of Systems and Software (JSS)

Olivier Barais

- International Journal on Software and Systems Modeling (SoSyM)
- Journal of Systems and Software
- Transactions on Software Engineering (TSE)

10.1.4. Invited talks

Benoit Baudry:

- MIT CSAIL: Automatic generation of software diversity in application source code
- LORIA (Nancy, France): Automatic software diversification in application code.

Benoit Combemale:

- McGill: From Model (driven) Engineering, to Language (driven) Engineering.
- IRT Saint Exupery: On the Globalization of Modeling Languages.
- INRA Toulouse: Modeling for Sustainability.

Jean-Marc Jézéquel:

- ECNU Shanghai: Model Driven Enginnering of DSLs
- Present And Ulterior Software Engineering Seminar (Chair of Software Engineering at ETH Zurich): Language Families

Olivier Barais:

- ECNU Shanghai: Model Driven Enginnering of DSLs
- EJCP 2015: Ecole Jeunes Chercheurs en Programmation

10.1.5. Leadership within the scientific community

Benoit Baudry:

- Steering committee member for the ACM/IEEE MODELS conference

Benoit Combemale:

- Steering committee member for the ACM SLE conference

10.1.6. Scientific expertise

Benoit Baudry:

- Member of the Scientific Advisory Board for the SVV lab, University of Luxembourg.

Arnaud Blouin:

- Member of the advisory board of the FNR project MODELS, Luxembourg.

Jean-Marc Jézéquel:

- Member of the Scientific Committee of the GDR GPL of CNRS.

10.1.7. Research administration

Jean-Marc Jézéquel is Director of IRISA (UMR 6074). He is Administrator of the IRT B-com (representing University of Rennes 1). He is Coordinator of the academic club of the French Cyber-defense Excellence Cluster, and Director of the Rennes Node of EIT Digital.

10.2. Teaching - Supervision - Juries

10.2.1. 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, CentraleSupelec Rennes, and ENSAI.

Noël Plouzeau is the manager 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.

10.2.2. Supervision

- HDR: Benoit Combemale: *Towards Language-Oriented Modeling*, 2015
- PhD: 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: Valéria Lelli, *On Testing Interactive Systems*, thèse 2012-2015, B. Baudry and A. Blouin
- PhD: Inti Gonzalez, *Ressources reservation in Pervasive Middleware*, thèse 2012-2015, J. Bourcier, O. Barais
- PhD: Ivan Paez Anaya, *Proactive Adaptation in Pervasive Environment*, thèse 2012-2015, J. Bourcier, N. Plouzeau and J.-M. Jézéquel
- PhD in progress : Sana Ben Nasr, *Modeling variability in regulatory requirements*, thèse 2013-2016, B. Baudry and M. Acher
- PhD in progress : Kwaku Yeboah-Antwi, *Runtime emergence of software diversity*, thèse 2013-2016, B. Baudry and O. Barais
- 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 : Paul Temple, *Massive Benchmarking of Vision Algorithms Using Synthetic Video Variants*, thèse 2015-2018, J.-M. Jézéquel and M. Acher
- PhD in progress : Quentin Plazar, *Combining Decision Procedures for Constraint Programming and SMT*, thèse 2015-2018, A. Gotlieb, S. Bardin and M. Acher
- PhD in progress : Mohamed Boussaa, *An Architecture for Testing Large-Scale Dynamic Distributed Systems*, thèse 2013-2016, O. Barais, G. Sunye, B. Baudry.
- 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 : 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

- PhD in progress : Marcelino Rodriguez-Cancio, *Automatic computation diversification*, thèse 2015-2018, B. Baudry and B. Combemale
- PhD in progress : Gwendal Le Moulec, *Towards synthesizing families of virtual reality applications*, thèse 2015-2018, with the Inria Hybrid Team, B. Arnaldi, A. Blouin, V. Gouranton

10.2.3. Juries

Benoit Baudry was in the examination committee of the following PhD thesis and HDR:

- Dionysios Efstathiou, October 2015, King's College London, Referee
- Aminata Sabané, December 2015, École Polytechnique de Montréal, Referee
- Mehdi Amhed-Nacer, May 2015, University of Nancy, Referee
- Mathieu Foucault, November 2015, University of Bordeaux, Examiner
- Erwan Bousse, December 2015, University of Rennes, Supervisor
- Valeria Lelli, December 2015, INSA Rennes, Supervisor
- Benoit Combemale, December 2015, University of Rennes, member

Benoit Combemale was in the examination committee of the following PhD thesis:

- Paola Vallejo, December 2015, University of Brest, Examiner
- Erwan Bousse, December 2015, University of Rennes, Supervisor

Arnaud Blouin was in the examination committee of the following PhD thesis:

- Valeria Lelli, December 2015, INSA Rennes, Supervisor

Jean-Marc Jézéquel was in the examination committee of the following PhD thesis and HDR:

- Jose Galindo Duarte, March 2015, University of Sevilla, Examiner
- Christopher Henard, May 2015, University of Luxembourg, Examiner
- Ivan Paez Anaya, September 2015, University of Rennes, Supervisor
- Remi Douence (HDR), January 2015, University of Nantes, Examiner
- Philippe Merle (HDR), September 2015, University of Lille, Examiner
- Axel Legay (HDR), November 2015, University of Rennes, President
- Benoit Combemale (HDR), December 2015, University of Rennes, Examiner
- Gerson Sunye (HDR), December 2015, University of Nantes, Examiner
- Jean-Rémy Falleri (HDR), December 2015, University of Bordeaux, Examiner
- Reda Bendraou (HDR), December 2015, University of Paris 6, Examiner

Johann Bourcier was in the examination committee of the following PhD :

- Ivan Paez Anaya, September 2015, University of Rennes, Supervisor
- Inti Gonzalez, December 2015, University of Rennes, Supervisor

Olivier Barais was in the examination committee of the following PhD :

- Inti Gonzalez, December 2015, University of Rennes, Supervisor
- Simon Urli, February 2015, University of Nice, Referee
- Sam Rottenberg, April 2015, Telecom Sud Paris, Referee
- Pierre Samson, November 2015, University of Pau, Referee

10.3. Popularization

Benoit Combemale participated to the "Inria Industry meeting" focused on "Power Transition", as part of the Futur en Seine festival. He presented the results of the collaboration with Obeo and INRA on the use of domain-specific languages for resource management.

Our research about browser fingerprinting have appeared in the MISC magazine (a French, general-audience magazine about technology and cyber-security) [75] and have been showcased at ICT'15 (a large European forum that gathers EU policy makers, industry and academics in the area of ICT) and at IRISA's open house.

11. Bibliography

Major publications by the team in recent years

[1] 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", 2015, vol. 32, n° 1, pp. 83–90, https://hal.archives-ouvertes.fr/hal-01089268

[2] 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° 1, https://hal.inria.fr/inria-00554197

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

[4] J.-M. DAVRIL, E. DELFOSSE, N. HARIRI, M. ACHER, J. CLELANG-HUANG, P. HEYMANS. *Feature Model Extraction from Large Collections of Informal Product Descriptions*, in "European Software Engineering Conference and the ACM SIGSOFT Symposium on the Foundations of Software Engineering (ESEC/FSE'13)", Saint Petersburg, Russia, September 2013, pp. 290-300 [*DOI :* 10.1145/2491411.2491455], https://hal.inria.fr/hal-00859475

[5] F. FOUQUET, E. DAUBERT, N. PLOUZEAU, O. BARAIS, J. BOURCIER, J.-M. JÉZÉQUEL. *Dissemination of reconfiguration policies on mesh networks*, in "DAIS 2012", Stockholm, Sweden, June 2012, https://hal.inria.fr/hal-00688707

[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] P.-A. MULLER, F. FLEUREY, J.-M. JÉZÉQUEL. *Weaving executability into object-oriented meta-languages*, in "Proceedings of MODELS/UML'2005", Montego Bay, Jamaica, October 2005, https://hal.inria.fr/hal-00795095

Publications of the year

Doctoral Dissertations and Habilitation Theses

[9] E. BOUSSE. *Execution Trace Management to Support Dynamic V&V for Executable DSMLs*, Rennes 1, December 2015, https://hal.inria.fr/tel-01238005

[10] B. COMBEMALE. *Towards Language-Oriented Modeling*, Université de Rennes 1, December 2015, Habilitation à diriger des recherches, https://hal.inria.fr/tel-01238817

[11] J. Á. GALINDO DUARTE. *Evolution, testing and configuration of variability systems intensive*, Université Rennes 1, March 2015, https://tel.archives-ouvertes.fr/tel-01187958

[12] I. GONZALEZ-HERRERA. *Supporting resource-awareness in managed runtime environments*, Universite Rennes 1, December 2015, https://hal.archives-ouvertes.fr/tel-01246035

[13] V. LELLI. *Testing and maintenance of graphical user interfaces*, INSA Rennes, November 2015, https://hal.archives-ouvertes.fr/tel-01232388

[14] I. D. PAEZ ANAYA. *Integrating predictive analysis in self-adaptive pervasive systems*, Université Rennes 1, September 2015, https://tel.archives-ouvertes.fr/tel-01251551

Articles in International Peer-Reviewed Journals

[15] 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", 2015, vol. 32, n^{o} 1, pp. 83–90, https://hal.archives-ouvertes.fr/hal-01089268

[16] 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", August 2015, vol. 14, n^{o} 3, pp. 1:1-43 [*DOI :* 10.5381/JOT.2015.14.3.A1], https://hal.inria.fr/hal-01083759

[17] B. BAUDRY, M. MONPERRUS. *The Multiple Facets of Software Diversity: Recent Developments in Year 2000 and Beyond*, in "ACM Computing Surveys", July 2015, 25 p. , https://hal.inria.fr/hal-01182103

[18] A. BLOUIN, B. COMBEMALE, B. BAUDRY, O. BEAUDOUX. *Kompren: Modeling and Generating Model Slicers*, in "Software and Systems Modeling", 2015, vol. 14, n^{o} 1, pp. 321-337, https://hal.inria.fr/hal-00746566

[19] A. BLOUIN, N. MOHA, B. BAUDRY, H. SAHRAOUI, J.-M. JÉZÉQUEL. *Assessing the Use of Slicing-based Visualizing Techniques on the Understanding of Large Metamodels*, in "Information and Software Technology", 2015, vol. 62, n^{o} 0, pp. 124 - 142 [*DOI :* 10.1016/J.INFSOF.2015.02.007], https://hal.inria.fr/hal-01120558

[20] 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

[21] J. CADAVID, B. COMBEMALE, B. BAUDRY. *An Analysis of Metamodeling Practices for MOF and OCL*, in "Computer Languages, Systems and Structures", 2015, vol. 41, 46 p. [*DOI :* 10.1016/J.CL.2015.02.002], https://hal.inria.fr/hal-01186015

[22] W. SUN, B. COMBEMALE, R. B. FRANCE, A. BLOUIN, B. BAUDRY, I. RAY. *Using Slicing to Improve the Performance of Model Invariant Checking*, in "Journal of Object Technology", 2015, 28 p. , https://hal.inria.fr/hal-01179369

International Conferences with Proceedings

[23] C. BOUDJENNAH, B. COMBEMALE, D. EXERTIER, S. LACRAMPE, M.-A. PERALDI-FRATI. *CLARITY: Open-Sourcing the Model-Based Systems Engineering Solution Capella*, in "Second Workshop on Open Source Software for Model Driven Engineering (OSS4MDE'15)", Ottawa, Canada, CEUR, 2015, https://hal.inria.fr/hal-01186019

[24] E. BOUSSE, T. MAYERHOFER, B. COMBEMALE, B. BAUDRY. *A Generative Approach to Define Rich Domain-Specific Trace Metamodels*, in "11th European Conference on Modelling Foundations and Applications (ECMFA)", L'Aquila, Italy, July 2015, https://hal.inria.fr/hal-01154225

[25] J.-M. BRUEL, B. COMBEMALE, I. OBER, H. RAYNAL. *MDE in Practice for Computational Science*, in "International Conference on Computational Science", Reykjavík, Iceland, June 2015, https://hal.inria.fr/hal-01141393

[26] B. COMBEMALE, C. BRUN, J. CHAMPEAU, X. CRÉGUT, J. DEANTONI, J. LE NOIR. *A Tool-Supported Approach for Concurrent Execution of Heterogeneous Models*, in "8th European Congress on Embedded Real Time Software and Systems (ERTS 2016)", Toulouse, France, 2016, https://hal.inria.fr/hal-01258358

[27] J.-M. DAVRIL, M. CORDY, P. HEYMANS, M. ACHER. *Using fuzzy modeling for consistent definitions of product qualities in requirements*, in "Artificial Intelligence for Requirements Engineering (AIRE), 2015 IEEE Second International Workshop on", Otawa , Canada, August 2015 [*DOI :* 10.1109/AIRE.2015.7337624], https://hal.inria.fr/hal-01243006

[28] 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

[29] T. DEGUEULE, B. COMBEMALE, A. BLOUIN, O. BARAIS. *Reusing Legacy DSLs with Melange*, in "15th Workshop on Domain-Specific Modeling", Pittsburgh, United States, Proceedings of the 15th Workshop on Domain-Specific Modeling, October 2015, https://hal.inria.fr/hal-01197039

[30] T. DEGUEULE, B. COMBEMALE, A. BLOUIN, O. BARAIS, J.-M. JÉZÉQUEL. *Melange: A Meta-language for Modular and Reusable Development of DSLs*, in "8th International Conference on Software Language Engineering (SLE)", Pittsburgh, United States, October 2015, https://hal.inria.fr/hal-01197038

[31] T. DEGUEULE, J. B. FERREIRA FILHO, O. BARAIS, M. ACHER, J. LE NOIR, S. MADELÉNAT, G. GAILLIARD, G. BURLOT, O. CONSTANT. *Tooling Support for Variability and Architectural Patterns in Systems Engineering*, in "19th International Conference on Software Product Line", Nashville, United States, Proceedings of the 19th International Conference on Software Product Line, July 2015 [*DOI :* 10.1145/2791060.2791097], https://hal.inria.fr/hal-01242180

[32] F. JOUAULT, O. BEAUDOUX, M. BRUN, M. CLAVREUL, G. SAVATON. *Towards Functional Model Transformations with OCL*, in "Proceedings the 8th International Conference on Model Transformation (ICMT'15)", L'Aquila, Italy, July 2015, pp. 111-120 [*DOI :* 10.1007/978-3-319-21155-8_9], https://hal.archives-ouvertes.fr/hal-01179294

[33] P. LAPERDRIX, W. RUDAMETKIN, B. BAUDRY. *Mitigating browser fingerprint tracking: multi-level reconfiguration and diversification*, in "Proceedings of the IEEE/ACM 10th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS)", Firenze, Italy, May 2015, pp. 98-108, https://hal.inria.fr/hal-01121108

[34] F. LATOMBE, X. CRÉGUT, B. COMBEMALE, J. DEANTONI, M. PANTEL. *Weaving Concurrency in eXecutable Domain-Specific Modeling Languages*, in "8th ACM SIGPLAN International Conference on Software Language Engineering (SLE)", Pittsburg, United States, ACM, 2015, https://hal.inria.fr/hal-01185911

[35] F. LATOMBE, X. CRÉGUT, J. DEANTONI, M. PANTEL, B. COMBEMALE. *Coping with Semantic Variation Points in Domain-Specific Modeling Languages*, in "1st International Workshop on Executable Modeling (EXE'15), co-located with MODELS'15", Ottawa, Canada, CEUR, 2015, https://hal.inria.fr/hal-01222999

[36] V. LELLI, A. BLOUIN, B. BAUDRY, F. COULON. *On Model-Based Testing Advanced GUIs*, in "11th Workshop on Advances in Model Based Testing (A-MOST 2015)", Graz, Austria, Software Testing, Verification and Validation Workshops (ICSTW), 2015 IEEE Eighth International Conference on, April 2015, pp. 1-10 [*DOI :* 10.1109/ICSTW.2015.7107403], https://hal.inria.fr/hal-01123647

[37] 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, pp. 1-10 [*DOI :* 10.1109/ICST.2015.7102582], https://hal.inria.fr/hal-01114724

[38] P. MERLE, O. BARAIS, J. PARPAILLON, N. PLOUZEAU, S. TATA. *A Precise Metamodel for Open Cloud Computing Interface*, in "8th IEEE International Conference on Cloud Computing (CLOUD 2015)", New York, United States, Proceedings of 8th IEEE International Conference on Cloud Computing (CLOUD 2015), IEEE, June 2015, pp. 852 - 859 [*DOI :* 10.1109/CLOUD.2015.117], https://hal.archives-ouvertes.fr/hal-01188800

[39] J.-M. MOTTU, S. SEN, J. CADAVID, B. BAUDRY. *Discovering Model Transformation Pre-conditions using Automatically Generated Test Models*, in "IEEE International Symposium on Software Reliability Engineering, ISSRE 2015", Washington DC, United States, November 2015, https://hal.archives-ouvertes.fr/hal-01228715

[40] E. OUTIN, J.-E. DARTOIS, O. BARAIS, J.-L. PAZAT. *Enhancing Cloud Energy Models for Optimizing Datacenters Efficiency*, in "IEEE International Conference on Cloud and Autonomic Computing (ICCAC)", Cambridge, MA, United States, IEEE, September 2015, 8 p. [*DOI :* 10.1109/ICCAC.2015.10], https://hal.inria.fr/hal-01243146

[41] J. PARPAILLON, P. MERLE, O. BARAIS, M. DUTOO, F. PARAISO. *OCCIware - A Formal and Tooled Framework for Managing Everything as a Service*, in "Projects Showcase @ STAF'15", L'Aquila, Italy, CEUR (editor), Proceedings of the Projects Showcase @ STAF'15, July 2015, vol. 1400, pp. 18 - 25, https://hal.archives-ouvertes.fr/hal-01188826

[42] W. SUN, B. COMBEMALE, R. B. FRANCE. *Towards the use of slicing techniques for an efficient invariant checking*, in "MODULARITY 2015", Fort Collins, United States, Companion Proceedings of the 14th International Conference on Modularity, MODULARITY 2015, Fort Collins, CO, USA, March 16 - 19, 2015, March 2015, 2 p. , Best poster award at Modularity'15 [*DOI :* 10.1145/2735386.2735926], https://hal.inria.fr/hal-01141395

[43] M. E. VARA LARSEN, J. DEANTONI, B. COMBEMALE, F. MALLET. *A Behavioral Coordination Operator Language (BCOoL)*, in "International Conference on Model Driven Engineering Languages and Systems (MODELS)", Ottawa, Canada, T. LETHBRIDGE, J. CABOT, A. EGYED (editors), ACM, September 2015, n° 18, 462 p. , to be published in the proceedings of the Models 2015 conference, https://hal.inria.fr/hal-01182773

Conferences without Proceedings

[44] M. ACHER, G. BÉCAN, B. COMBEMALE, B. BAUDRY, J.-M. JÉZÉQUEL. *Product Lines Can Jeopardize Their Trade Secrets*, in "10th Joint Meeting of the European Software Engineering Conference and the ACM SIGSOFT Symposium on the Foundations of Software Engineering", Bergame, Italy, August 2015 [*DOI :* 10.1145/2786805.2803210], https://hal.inria.fr/hal-01234342

[45] M. ACHER, R. E. LOPEZ-HERREJON, R. RABISER. *SPLTea 2015: Second International Workshop on Software Product Line Teaching*, in "19th International Conference on Software Product Line (SPLC'15)", Nashville, United States, Proceedings of the 19th International Conference on Software Product Line (SPLC'15), July 2015 [*DOI :* 10.1145/2791060.2791063], https://hal.inria.fr/hal-01243213

[46] S. BEN NASR, G. BÉCAN, M. ACHER, J. B. FERREIRA FILHO, B. BAUDRY, N. SANNIER, J.-M. DAVRIL. *MatrixMiner: A Red Pill to Architect Informal Product Descriptions in the Matrix*, in "10th Joint Meeting of the European Software Engineering Conference and the ACM SIGSOFT Symposium on the Foundations of Software Engineering", Bergame, Italy, August 2015 [*DOI :* 10.1145/2786805.2803180], https://hal.inria.fr/hal-01234338

[47] M. BOUSSAA, O. BARAIS, G. SUNYÉ, B. BAUDRY. *A Novelty Search Approach for Automatic Test Data Generation*, in "8th International Workshop on Search-Based Software Testing SBST@ICSE 2015", Firenze, Italy, May 2015, 4 p. , https://hal.archives-ouvertes.fr/hal-01121228

[48] M. BOUSSAA, O. BARAIS, G. SUNYÉ, B. BAUDRY. *A Novelty Search-based Test Data Generator for Object-oriented Programs*, in "GECCO 2015", Madrid, Spain, Proceedings of the Companion Publication of the 2015 Annual Conference on Genetic and Evolutionary Computation, ACM, July 2015, pp. 1359–1360 [*DOI :* 10.1145/2739482.2764716], https://hal.archives-ouvertes.fr/hal-01248177

[49] E. BOUSSE, J. CORLEY, B. COMBEMALE, J. GRAY, B. BAUDRY. *Supporting Efficient and Advanced Omniscient Debugging for xDSMLs*, in "8th International Conference on Software Language Engineering (SLE)", Pittsburg, United States, October 2015, https://hal.inria.fr/hal-01182517

[50] 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

[51] G. BÉCAN, R. BEHJATI, A. GOTLIEB, M. ACHER. *Synthesis of Attributed Feature Models From Product Descriptions*, in "International Software Product Line Conference", Nashville, United States, July 2015, https://hal.inria.fr/hal-01178454

[52] B. COMBEMALE, J. DEANTONI, O. BARAIS, A. BLOUIN, E. BOUSSE, C. BRUN, T. DEGUEULE, D. VOJTISEK. *A Solution to the TTC'15 Model Execution Case Using the GEMOC Studio*, in "8th Transformation Tool Contest", l'Aquila, Italy, CEUR, 2015, https://hal.inria.fr/hal-01152342

[53] J.-M. DAVRIL, P. HEYMANS, G. BÉCAN, M. ACHER. *On Breaking The Curse of Dimensionality in Reverse Engineering Feature Models*, in "17th International Configuration Workshop", Vienna, Austria, 17th International Configuration Workshop, September 2015, vol. 17th International Configuration Workshop, https://hal.inria.fr/hal-01243571

[54] J. B. FERREIRA FILHO, M. ACHER, O. BARAIS. *Challenges on Software Unbundling: Growing and Letting Go*, in "Modularity'15", Fort Collins, CO, United States, March 2015, https://hal.inria.fr/hal-01116694

[55] F. FLEUREY, B. BAUDRY, B. GAUZENS, A. ELIE, K. YEBOAH-ANTWI. *Emergent Robustness in Software Systems through Decentralized Adaptation: an Ecologically-Inspired ALife Approach*, in "European Conference on Artificial Life 2015", York, United Kingdom, July 2015, https://hal.inria.fr/hal-01159131

[56] A. MOAWAD, T. HARTMANN, F. FOUQUET, G. NAIN, J. KLEIN, J. BOURCIER. *Polymer: A Model-Driven Approach for Simpler, Safer, and Evolutive Multi-Objective Optimization Development*, in "International Conference on Model-Driven Engineering and Software Development", Angers, France, February 2015, https://hal.inria.fr/hal-01248160

[57] S. PINCHINAT, M. ACHER, D. VOJTISEK. *ATSyRa: An Integrated Environment for Synthesizing Attack Trees* , in "Second International Workshop on Graphical Models for Security (GraMSec'15) co-located with CSF'15", Verona, Italy, July 2015, https://hal.inria.fr/hal-01243021

[58] J. L. RODAS, D. MENDEZ ACUNA, J. A. GALINDO DUARTE, D. BENAVIDES, J. CARDENAS. *Towards testing variability intensive systems using user reviews*, in "Congreso Colombiano de Computación", Bogotá, Colombia, Rubby Casallas, September 2015, https://hal.inria.fr/hal-01204507

[59] M. E. VARA LARSEN, J. DEANTONI, B. COMBEMALE, F. MALLET. *A Model-Driven Based Environment for Automatic Model Coordination*, in "Models 2015 demo and posters", Ottawa, Canada, CEUR (editor), Models 2015 demo and posters, October 2015, https://hal.inria.fr/hal-01198744

[60] K. YEBOAH-ANTWI, B. BAUDRY. *Embedding Adaptivity in Software Systems using the ECSELR framework*, in "GECCO'15", Madrid, Spain, July 2015, https://hal.inria.fr/hal-01159118

Scientific Books (or Scientific Book chapters)

[61] *Joint Proceedings of the 3rd International Workshop on the Globalization Of Modeling Languages and the 9th International Workshop on Multi-Paradigm Modeling*, CEUR, Ottawa, Canada, 2015, vol. 1511, https://hal.inria.fr/hal-01242558

[62] B. BRYANT, J.-M. JÉZÉQUEL, R. LAEMMEL, M. MERNIK, M. SCHINDLER, F. STEINMANN, J.-P. TOLVANEN, A. VALLECILLO, M. VOELTER. *Globalized Domain Specific Language Engineering*, in "Globalizing Domain-Specific Languages", B. COMBEMALE, B. H. CHENG, R. B. FRANCE, J.-M. JÉZÉQUEL, B. RUMPE (editors), Springer International Publishing, October 2015, vol. 9400, pp. 43-69 [*DOI :* 10.1007/978-3-319-26172-0_4], https://hal.inria.fr/hal-01224283

[63] B. H. CHENG, B. COMBEMALE, R. B. FRANCE, J.-M. JÉZÉQUEL, B. RUMPE. *On the Globalization of Domain-Specific Languages*, in "Globalizing Domain-Specific Languages", LNCS, Springer International Publishing, 2015, vol. 9400 [*DOI :* 10.1007/978-3-319-26172-0_1], https://hal.inria.fr/hal-01224105

[64] B. H. CHENG, T. DEGUEULE, C. ATKINSON, S. CLARKE, U. FRANK, P. J. MOSTERMAN, J. SZTIPANOVITS. *Motivating Use Cases for the Globalization of DSLs*, in "Globalizing Domain-Specific Languages", B. COMBEMALE, B. H. CHENG, R. B. FRANCE, J.-M. JÉZÉQUEL, B. RUMPE (editors), Lecture Notes in Computer Science, Springer International Publishing, 2015, vol. 9400, pp. 21-42 [*DOI :* 10.1007/978-3-319-26172-0_3], https://hal.inria.fr/hal-01233660

[65] T. CLARK, M. VAN DEN BRAND, B. COMBEMALE, B. RUMPE. *Conceptual Model of the Globalization for Domain-Specific Languages*, in "Globalizing Domain-Specific Languages", B. COMBEMALE, B. H. CHENG, R. B. FRANCE, J.-M. JÉZÉQUEL, B. RUMPE (editors), Lecture Notes in Computer Science, Springer International Publishing, 2015, vol. 9400, pp. 7-20 [*DOI :* 10.1007/978-3-319-26172-0_2], https://hal.inria.fr/hal-01224116

[66] B. COMBEMALE, B. H. CHENG, R. B. FRANCE, J.-M. JEZEQUEL, B. RUMPE. *Globalizing Domain-Specific Languages*, LNCS, Programming and Software Engineering, Springer International Publishing, 2015, vol. 9400 [*DOI :* 10.1007/978-3-319-26172-0], https://hal.inria.fr/hal-01224096

Research Reports

[67] G. BÉCAN, R. BEHJATI, A. GOTLIEB, M. ACHER. *Synthesis of Attributed Feature Models From Product Descriptions: Foundations*, Inria Rennes ; Inria, February 2015, n^{o} RR-8680, https://hal.inria.fr/hal-01116663

Scientific Popularization

[68] J. B. FERREIRA FILHO, S. ALLIER, O. BARAIS, M. ACHER, B. BAUDRY. *Assessing Product Line Derivation Operators Applied to Java Source Code: An Empirical Study*, in "19th International Software Product Line Conference (SPLC'15)", Nashville, TN, United States, July 2015, https://hal.inria.fr/hal-01163423

[69] F. PARAÏSO, J. PARPAILLON, P. MERLE. *Model-Driven Multi-Cloud Resource Management*, October 2015, EIT Digital Future Cloud Symposium présenté, Poster, https://hal.inria.fr/hal-01218571

Other Publications

[70] B. BAUDRY, S. ALLIER, M. RODRIGUEZ-CANCIO, M. MONPERRUS. *Automatic Software Diversity in the Light of Test Suites*, September 2015, 11 pages, 4 figures, 8 listings, conference, https://hal.archives-ouvertes.fr/hal-01198384

[71] B. BAUDRY, S. ALLIER, M. RODRIGUEZ-CANCIO, M. MONPERRUS. *DSpot: Test Amplification for Automatic Assessment of Computational Diversity*, June 2015, 11 pages, https://hal.archives-ouvertes.fr/hal-01162219

[72] B. COMBEMALE, B. H. CHENG, A. MOREIRA, J.-M. BRUEL, J. GRAY. *Modeling for Sustainability*, 2015, working paper or preprint, https://hal.inria.fr/hal-01185800

[73] T. DEGUEULE. *Towards Language Interfaces for DSLs Integration*, March 2015, working paper or preprint, https://hal.inria.fr/hal-01138017

[74] J. H. KIM, A. LEGAY, L.-M. TRAONOUEZ, M. ACHER, S. KANG. *A Formal Modeling and Analysis Framework for Software Product Line of Preemptive Real-Time Systems*, October 2015, Publication acceptée en temps que poster/papier court à la conférence SAC 2016, section software engineering, https://hal.archives-ouvertes.fr/hal-01241673

[75] P. LAPERDRIX, B. BAUDRY. *Le fingerprinting : une nouvelle technique de traçage*, September 2015, pp. 52-57, MISC Magazine n°81, https://hal.inria.fr/hal-01247090

[76] J. XUAN, B. CORNU, M. MARTINEZ, B. BAUDRY, L. SEINTURIER, M. MONPERRUS. *Dynamic Analysis can be Improved with Automatic Test Suite Refactoring*, June 2015, working paper or preprint, https://hal.archives-ouvertes.fr/hal-01162220

References in notes

[77] 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

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

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

[80] 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° 4, pp. 45–52

[81] 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

[82] 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° 1, pp. 3–40

[83] 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

[84] 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° 1, pp. 3–22

[85] 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

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

[87] 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

[88] 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

[89] L. C. 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° 4, pp. 387–404

[90] 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

[91] 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

[92] 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

[93] 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

[94] 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

[95] 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 Heidelberg, Berlin, Heidelberg, 2009, vol. 5525

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

[97] 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

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

[99] 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

[100] 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

[101] 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

[102] R. B. 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

[103] 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

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

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

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

[107] 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

[108] 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

[109] 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

[110] 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

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

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

[113] 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

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

[115] 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

[116] 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

[117] 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

[118] 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.

[119] 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

[120] 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

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

[122] L. M. 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

[123] 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

[124] E. OUTIN, J.-L. PAZAT, O. BARAIS. *Using Models@Run.time to embed an Energetic Cloud Simulator in a MAPE-K Loop*, in "Workshop Autonomique", Toulouse, France, October 2014, https://hal.inria.fr/hal-01243158

[125] 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

[126] 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

[127] 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

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

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

[130] 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

[131] 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

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

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

[134] 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

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

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

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

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

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

[140] 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^{o} 1, pp. 20–34

[141] Z. A. ZHU, S. MISAILOVIC, J. A. KELNER, M. 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

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