Evaluation of Inria theme Distributed programming and Software engineering

Project-team DiverSE

https://www.diverse-team.fr

October 2021

Project-Team Title: Diversity-centric Software Engineering DiverSE

Theme: Distributed programming and Software engineering

Scientific Leader: Benoit Baudry - Olivier Barais

Research Center: Rennes – Bretagne Atlantique

Common Project-Team with: University of Rennes 1, CNRS, INSA Rennes

DiverSE is located on Beaulieu campuses: Rennes

Abstract: 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 and Instrumentation and Control to pervasive combinations of Internet of Things and Internet of Services. 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 dimension of diversity: the diversity of languages used by the stakeholders involved in the construction of these systems; the diversity of features (aka variability) 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 envelope

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

DiverSE long-term ambition

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.

1 Personnel

Current Composition of the Project-Team:

Research Scientists and Faculty Members:
- Mathieu Acher, Associate Professor, Univ de Rennes I
- Olivier Barais, Team leader, Full Professor, Univ de Rennes I
- Arnaud Blouin, Associate Professor, HDR, INSA Rennes
- Johann Bourcier, Associate Professor, HDR, Univ de Rennes I
- Stéphanie Challita, Associate Professor, Univ de Rennes I, *from Sept 2020*
- Benoit Combemale, Full Professor, Univ de Rennes I
- Jean-Marc Jezequel, Full Professor, Univ de Rennes I
- Djamel Eddine Khelladi, Full time Researcher, CR CNRS, *from January 2019*
- Noel Plouzeau, Associate Professor, Univ de Rennes I
- Olivier Zendra, Full time Researcher, CR INRIA, *from Oct 2020*

Research Engineers:
- Didier Vojtisek, Inria
- Romain Belafia, Univ de Rennes I
- Romain Lefeuvre, Univ de Rennes I
- Bruno Lebon, INRIA

Post-docs:
- Dorian Leroy, INRIA
- Xhevahire Tërnava, Univ de Rennes I
- Elyes Cherfa, Univ de Rennes I

PhD Students:
- Luc Lesoil, Univ Rennes I
- Alif Akbar Pranata, Inria
- June Benvegnu Sallou, Univ de Rennes I
- Emmanuel Chebbi, Inria
- Antoine Cheron, FaberNovel, granted by CIFRE
- Fabien Coulon, Obeo, granted by CIFRE
- Pierre Jeanjean, Inria
- Gauthier Lyan, Keolis Rennes, granted by CIFRE
- Hugo Martin, Univ de Rennes I
- Anne Bumiller, Orange, granted by CIFRE
- Cassius De Oliveira Puodzius, Inria
- Gwendal Jouneaux, Univ de Rennes I
- Quentin Le Dilavrec, Univ de Rennes I
- Georges Aaron Randrianaina, Univ de Rennes I

Administrative Assistants:
- Sophie Maupile, CNRS

External Collaborators:
- Gurvan Le Guernic, DGA

Personnel at the Start of the Evaluation Period (October 2016)

	Inria	CNRS	University	Other	**Total**
DR (1) / Professors			2		**2**
CR (2) / Associate professors	1		5		**6**
Permanent engineers	1				**1**
Temporary engineers	2	1	2		**5**
Post-docs	1		2		**3**
PhD Students	3	2	7	2	**14**
Total	**8**	**4**	**17**	**2**	**31**

(1) "Senior Research Scientist (Directeur de Recherche)"
(2) "Junior Research Scientist (Chargé de Recherche)"

Personnel at the Time of the Evaluation (October 2021)

	Inria	CNRS	University	Other	**Total**
DR / Professors			3		**3**
CR / Associate professors	1	1	5		**7**
Permanent engineers	1				**1**
Temporary engineers	1		2		**3**
Post-docs	1		2		**3**
PhD Students	4		6	4	**14**
Total	**8**	**1**	**18**	**4**	**31**

Changes in the Scientific Staff

	Inria	CNRS	University	Other	**Total**
DR / Professors			+1		**+1**
CR / Assistant Professors		+1			**+1**

Current Position of Former Project-Team Members

- Caroline Landry, Software Development Manager at MediaKind, Rennes France
- Juliana Alves Pereira, Univ de Rennes I
- Erwan Bousse, Associate Professor at University of Nantes, France
- Oscar Luis Vera Perez, Research Engineer at MediaKind, Rennes France
- Youssou Ndiaye: Security Engineer at Famoco, Rennes, France
- Ludovic Mouline, Software Engineer at SHARE NOW, Luxembourg, Luxembourg
- Manuel Leduc, Research Engineer at XWiki, Paris, France
- José Galindo, Professor at Univ de Sevilla, Seville, Spain
- David Méndez-Acuña, Software Engineer at Adobe, Paris, France
- Francisco Javier Acosta Padilla, Embedded Hardware and Software Engineering Manager at KUGU, Berlin, Germany
- Jacky Bourgeois, Assistant Professor, TU Delft, Delft, Netherlands
- Mohammed Boussaa, Software Engineer at ActiveEon, Paris, France
- Gwendal Le Moulec, Software Engineer at Sopra Steria, Rennes, France
- Kevin Corre, Research engineer at EXFO, Rennes, France
- Paul Temple, Postdoc at Université de Namur, Namur, Belgium
- Amine Benelallam, Senior Software Engineer at MathWorks, Nanterre, France
- Marcelino Rodriguez Cancio, Chief Technology Officer at Couture Technologies, Nashville, Tennessee, United States
- Johan Pelay, teacher at Université de Pau et des Pays de l'Adour, Pau, France

- Pierre Laperdrix, full time researcher at CNRS, Lille, France
- Jean-Emile Dartois, Research Engineer at Orange Labs, Rennes France
- Alexandre Rio, OKWind, Research Engineer at OkWind, Vitre France
- Thomas Degueule, full time researcher at CNRS, Bordeaux, France

Last Inria Enlistments

No new INRIA researcher has been recruited for the team during the period. Olivier Zendra, head of the Tamis team, joined the team in October 2020. Benoit Baudry, head of the DiverSE team until Sept 2017, moved to KTH Stockholm.

Comments:

The field of software engineering is at the crossroads of many fields. In particular, the proximity with the domain of system, distributed systems but also the numerous implications in the domain of software security imposes to have within the team many skills both on the technological aspects (technology in constant evolution) and in the domain of execution platforms, programming language, modeling, machine learning, statistics (empirical evaluation).

Following the move of Benoit Baudry to KTH Stockholm, Olivier Barais led the team from September 2017. In September 2017, Benoit Combemale took a full-professor position at the University of Toulouse. Jean-Marc Jézéquel was dean of IRISA (CNRS laboratories 1000 employees). Our priority was to quickly consolidate the team of permanent staff in order to be able to correctly supervise the many doctoral students who were present. For family reasons, Benoit Combemale had the opportunity to come back on a half-delegation to INRIA in September 2018 and to come definitely back to Rennes on a faculty position in September 2020. Since September 2020, he has informally assumed a role as scientific co-supervisor of the team. If the team is renewed, we will propose to INRIA to formalize this role. Djamel Eddine Khelladi was recruited as a tenured CNRS Research Scientist in January 2019, then Stéphanie Challita, a PhD student of the INRIA Spirals team, was recruited in September 2020 as a UR1 Associate Professor. Finally Olivier Zendra joined the team in October 2020. Jean-Marc Jézéquel is no longer director of IRISA since January 2021. In summary, in addition to the classic movements of many non-permanent staff during the period, there were also many arrivals and the departure of the initial team leader. The consolidation in terms of permanent staff has been mainly ensured by the CNRS and the University for the moment. There is no doubt that INRIA wishes to support the team in this growth but as in many fields, international competition is important for the recruitment of young scientists in the field.

Figure 1: The DIVERSE Group: A Software Engineering Group

2 Research Goals and Results

2.1 Keywords

Software Engineering: empirical software engineering, software maintenance & evolution, software language, software testing, domain-specific language, Software Architecture & Design, Dynamic reconfiguration, Web, Object-oriented programming, component-based design, middleware, Privacy-enhancing technologies, Security of equipment and software, resource management
Application Domains: software engineering, software industry, software evolution & maintenance, sustainable development, resource management, Internet of things, information systems, embedded systems, sensor networks for smart buildings, privacy, user interface engineering

2.2 Context and Overall Goals of the Project

DiverSE's research agenda targets core values of software engineering. In this fundamental domain we focus and develop models, methodologies and theories to address major challenges raised by the emergence of several forms of diversity in the design, deployment and evolution of software-intensive systems. Software diversity has emerged as an essential phenomenon in all application domains born by our industrial partners. These application domains range from complex systems brought by systems of systems (addressed in collaboration with Thales, Safran, CEA and DGA) and Instrumentation and Control (addressed with EDF) to pervasive combinations of Internet of Things and Internet of Services (addressed with TellU and Orange) and tactical information systems (addressed in collaboration with civil security). Today these systems seem to be radically all different, but we envision a strong convergence of the scientific principles that underpin their construction and validation, bringing forwards sane and reliable methods for the design of **flexible and open yet dependable systems**. Flexibility and openness are both critical and challenging software layer properties that must deal with the following four dimensions of diversity: **diversity of languages**, used by the stakeholders involved in the construction of these systems; **diversity of features**, required by the different customers; **diversity of runtime environments**, in which software has to run and adapt; **diversity of implementations**, which are necessary for resilience by redundancy.

In this context, the central 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 yet valid ways, to changes in the requirements and environment. Conversely, explicitly handling diversity is a great opportunity to allow software to spontaneously explore alternative design solutions. Concretely, we want to provide software engineers with the following abilities:

- 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 summarized below:

Scientific objective for DiverSE:

To automatically **compose and synthesize software diversity** from design to runtime to **address unpredictable evolution 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 this aspect stands as 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.

DiverSE's scientific background can be read as an appendix of this document.

2.3 Research axis

Figure 2 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 axes share and leverage the scientific and technological results developed in the area of model-driven engineering in the last decade. This means that all our research activities are founded on sound abstractions to reason about specific aspects of software systems, compose different perspectives and automatically generate parts of the system.

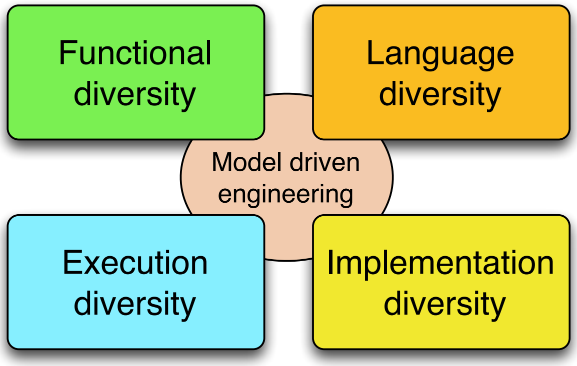

Figure 2: The four research axes of DiverSE, which rely on a MDE scientific background

2.4 Research Axis 1: Software Language Engineering

2.4.1 Personnel

Permanent fellows: M. Acher, O. Barais, B. Baudry A. Blouin, B. Combemale, J.-M. Jézéquel, D. Vojtisek.
Postdocs: J. A. Galindo, Elyes Cherfa.
PhD students: E. Bousse, F. Coulon, T. Degueule, P. Jeanjean, M. Leduc, D. Leroy, G. Le Moulec, D. Méndez-Acuña, G Jouneaux
Research engineers: R Belafia, R. Lefeuvre

2.4.2 Project-Team Positioning

The engineering of systems involves many 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 [Sch06, FR07]. 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.

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 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) to set principles for modeling language units that support the modular specification of a modeling language; and (ii) to design mechanisms to assemble these units into 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) gives 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.

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 support to 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. This framework 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 to MoCs and DSMLs design and implementation and to 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 [SDMHR11] (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.

2.4.3 Scientific Achievements

We developed our scientific contributions related to software language engineering through three axes: software language foundations; software language development approaches and Integrated Development Environments (IDE); software language applications. We developed in a book [190] these axes to guide engineers in handling MDE and SLE concepts and toolings.

Software Language Foundations We proposed in [39, 193] a disciplined approach that leverages type groups' polymorphism to provide an advanced type system for manipulating models, in a polymorphic way. This type system complements the Melange language workbench [DCB+15] and is seamlessly integrated into the Eclipse Modeling Framework (EMF) [SBMP08]. This research work is integrated within the Gemoc Studio [87], which is now an Eclipse project[1]. The PhD student involved in those work is one of the **2016 Grand Finals winners of the ACM's Student Research Competition**[2]. The research works on a new typing system of languages permitted to work on the problem of **composition, extension, and reuse** of software languages. We introduced the *Language Extension Problem* as a way to better qualify the scope of the challenges related to language extension and composition [52]. We believe the *Language Extension Problem* will drive future research in SLE, the same way the original *Expression*

[1] https://www.eclipse.org/org/research/project/gemoc_studio/
[2] https://www.acm.org/media-center/2016/june/src-2016-grand-finals

Problem helped to understand the strengths and weaknesses of programming languages. We also stipulated the existence of a unifying framework that reduces all structural composition operators to structural merging, and all composition operators acting on discrete behaviors to event scheduling [48]. This has been established on the basis of previous formal description of such operators [45]. We then proposed a new language implementation pattern, named REVISITOR, that enables independent extensibility of the syntax and semantics of metamodel-based software languages with incremental compilation and without anticipation [133]. Following, we introduced a non-intrusive approach to modular development of language concerns with well-defined interfaces that can be composed modularly at the specification and implementation levels [134] (**awarded by the best artefact award at SLE 2018**). The goal of modular language development is to enable the definition of new languages as assemblies of pre-existing ones. Regarding reuse, we designed a high-level classification on the ability of the current reuse approaches for model transformations across different metamodels [96, 28] (awarded by the **best paper award at ICMT 2018**), and introduced a conceptual framework to foster reuse in modeling [124]. Following a bottom-up approach, we also analysed existing software language artefacts to: enable reuse of software language artefacts through the definition of language modules which can be later put together to build up new DSLs [60]; design a methodology for isolating Wikipedia's articles about software languages and based on article's features [114]. Regarding executable DSLs, we propose a metalanguage for complementing the definition of executable DSLs with explicit behavioral interfaces to enable external tools to interact with executed models in a unified way [55]. We thus propose a runtime monitoring generic approach for executable DSLs [56], and more recently we propose a language-agnostic, unifying framework for runtime monitoring and logging, and demonstrate how it can be used to define loggers, runtime monitors and combinations of the two, aka. moniloggers [139]. We proposed a novel generative approach that defines a multidimensional and domain-specific trace metamodel enabling the construction and manipulation of execution traces for models conforming to a given xDSML [27]. We then proposed a set of formally defined operators for analyzing execution traces [137]. Regarding debugging, we: proposed a generic omniscient debugger supported by efficient generic trace management facilities [26]; identified temporal behavioral patterns that associate debugging activities with build success, which corroborates the positive impact of debugging practices on software development [175]. Regarding performance of executable DSL, little attention has been given to the performance of DSL interpreters. In [53], we systematically exploit the domain-specific information of language specifications to derive optimized Truffle-based language interpreters executed over the GraalVM. Following, we developed an approach for the concurrent execution of heterogeneous models [98]. The approach relies on an explicit model of concurrency and domain-specific actions with a well-defined protocol between them reified through explicit domain-specific events. We also investigate the concept of *self-adaptable languages* [119, 118]. While over recent years self-adaptation has become a concern for many software systems that have to operate in complex, software languages do not capitalize on monitored usage data of a language and its modeling environment, and do not use dedicated self-adaptation concerns. To address these issues, this paper introduces the concept of *Self-Adaptable Language* to abstract the feedback loops at both system and language levels.

Finally, we proposed a novel technique for processing user interactions [23] (at IEEE TSE). This novel technique permits the creation of highly interactive language environments.

Software Language development approaches and IDEs The novel concepts we detailed in the previous section consists of a common base for working on the next generations of DSL development approaches and DSL IDEs. Regarding software Language development approaches, in the new ideas paper [101], awarded as the **best new ideas paper at SLE 2018**, we envisioned a language engineering approach enabling (i) language users to manipulate language constructs in the most appropriate shape according to the task at hand, and (ii) language designers to combine the strengths of different technologies for a single DSL.

Regarding DSL IDEs, IDEs are evolving from monolithic desktop applications towards cloud-native applications with the aim to relocate the language services provided by an IDE on distant servers. We first conducted our experiment on the microservicization process of the Cloud-Based graphical modeling workbench Sirius Web [88]. We aimed to identify the technical challenges related to applications with similar properties, and provide insights for practitioners to migrate their similar applications towards microservices. In [100], we also explored the modularization of all language services to support their individual deployment and dynamic adaptation within an IDE. New languages appear regularly offering unique constructs and supported by dedicated services to be offered as new capabilities in IDEs. This trend leads to the multiplication of new protocols, hard to combine and possibly incompatible. We presented in [117] an approach that can make use of these specifications in order to deploy an IDE as a set of coordinated, individually deployed, language capabilities (e.g., microservice choreography). IDEs went from directly supporting languages to protocols, and we envision in this paper the next step: *IDE as Code*, where language protocols are created or inferred on demand and serve as support of an adaptation loop taking in charge of the (re)configuration of the IDE. In [116], we proposed an approach to automatically generate interactive computer programming environments from existing specifications of textual interpreted DSLs. The approach provides abstractions to complement the DSL specification, and combines static analysis and language transformations to automate the transformation of the language syntax, the execution state and the execution semantics. Closely related, in [180] we proposed an approach to REPL (Read-eval-print-loops) interpreters. The obtained REPLs can be generically turned into an exploring interpreter, to allow exploration of the user's interaction.

Provide practitioners with usable DSL environments is challenging. We first proposed an approach that focuses on easing the production of documentation of textual DSLs [51, 11]. We proposed a model-driven approach that relies on DSL artifacts to extract information required to build documentation. We also investigate live modeling, an important technique to edit behavioral models while being executed and helps in better understanding the impact of a design choice. We developed Live-UMLRT [85], a tool that supports live modeling of UML-RT models when they are executed by code generation. We also investigated modeling in the context of state machine models and code generation [20]. Finally, the ever-growing complexity of systems, the growing number of stakeholders, and the corresponding continuous emergence of new domain-specific modeling abstractions has led to significantly higher cognitive load on modelers. There is an urgent need to provide modelers with better modeling tools and processes. We developed roadmaps for: socio-technical coordination in the context of heterogeneous modeling [91] and of IoT systems [75]; intelligent modeling assistants [158, 61].

2.4.4 Software Language Applications

Application to Data-Centric Applications The ever increasing availability of data lead complex systems to integrate them all along the life cycle, from design to runtime. In [30], we introduce a conceptual reference framework – the Models and Data (MODA) framework – to support a data-centric and model-driven approach for the integration of heterogeneous models and their respective data for the entire life-cycle of socio-technical systems. We also explored the consideration of the time dimension, both for design [90] and for prediction [143, 142].

Application to DevOps and Digital Twins. Models open opportunities to offer a smooth continuum from design to runtime. In this context, we explored the opportunities for engineering digital twins [92], and to promote DevOps practices at the model level [191].

Application to Scientific Computing and Sustainability. Trading off some accuracy for better performances in scientific computing in general is an appealing approach to ease the exploration of various alternatives on complex simulation models. We proposed in [173] a new approximate computing technique, aka. loop aggregation, which consists in automatically reducing the main loop of a simulation model by aggregating the corresponding spatial or temporal data. Moreover, in [57], we investigate the different levels of abstraction, linked to the diverse artifacts of the scientific software development process, a software language can propose, and the V&V facilities associated to the corresponding level of abstraction the language can provide to the user. We aim to raise awareness among scientists, engineers and language providers on their shared responsibility in developing reliable scientific software. Finally, sustainability has emerged as a concern of central relevance. As a wicked problem, it poses challenges to business-as-usual in many areas, including that of modeling. In [47, 99] we address a question at the intersection of model-driven engineering and sustainability research: How can we better support sustainability by bringing together model-driven engineering, data, visualization and self-adaptive systems, to facilitate engagement, exploration, and understanding of the effects that individual and organizational choices have on sustainability?

Application to High-Performance Computing. In [54], we discuss the main lessons learned to be considered for conducting future projects in the field of HPC, and the remaining challenges that are worth being included in the road-map of the MDE community.

Application to Industry 4.0. Industry 4.0 integrates cyber-physical systems with the IoT to optimize the complete value-added chain. Successfully applying Industry 4.0 requires the cooperation of various stakeholders from different domains. We aim to assess the use of modeling in Industry 4.0 through the lens of modeling languages in a broad sense by conducting in [71] a systematic mapping study through which we developed an updated map of the research landscape on modeling languages and techniques for Industry 4.0.

Application to Virtual Reality. Due to the nature of Virtual Reality (VR) research, conducting experiments in order to validate the researcher's hypotheses is a must. However, the development of such experiments is a tedious and time-consuming task. In [131], we propose to make this task easier, more intuitive and faster with a method able to describe and generate the most tedious components of VR experiments.

Application to Games. In [138], we present an approach to complement an operational semantics for handling stimuli, and by automatically generating a complete behavioral language interface from this augmentation. We demonstrate how it can be used to implement a Pac-Man DSL enabling to create and play Pac-Man games.

Application to HPC. In [135], we introduce NabLab, a full-fledged industrial environment for scientific computing and High Performance Computing (HPC), involving several metamodels and grammars. Beyond the description of an industrial experience of the development and use of tool-supported DSLs, we report in this paper our lessons learned, and demonstrate the benefits from usefully combining metamodels and grammars in an integrated environment.

2.4.5 Collaborations

During this period, we worked with CEA (DSL for HPC), CWI (agility in language engineering), McGill (model reuse and self-adaptable languages), Aachen (MDE for industry 4.0), TU Wien (model executability), Lancaster University (MDE for environmental sciences), JKU Linz (co-evolution, and MDE for DevOps and digital twins), TU Eindhoven (MDE for DevOps and digital twins), Univ. L'alquila (MDE for DevOps and digital twins) and ETS Montreal (MDE for DevOps and digital twins). We also had a collaboration internally at Inria with the team Hybrid (usability in DSLs), and with the environmental sciences labs OSUR at University of Rennes 1.

2.4.6 External support

The work has been supported by bilateral collaboration with industry (Inria/Safran GLOSE, Inria/CEA Debug4Science, CIFRE Obeo, UR1/DGA Family, FPML and LangComponent), collaborative projects (IDM4SCO, OneWay, IPSCo, Clarity), international associate teams (ALE, Emma), and two French PhD grants (MESR).

2.4.7 Self Assessment

In this research axis, we conducted foundational and experimental research activities leading to publications in both top tier conferences and journals (*e.g.*, IEEE Software/Computer, Communicatin of the ACM, SoSym), as well as more specialized venues (*e.g.*, ACM SIGPLAN SLE or IEEE/ACM SIGSOFT MODELS). We developed a strong academic leaderships (deputy editor in chief at JOT, chair of the steering committee at SLE, associate editor at SOSyM, member of the steering committee at MODELS...), and provided large efforts to animate the community (*e.g.,* conferences and Dagstuhl/Bellairs workshops organisation). We also trained numerous PhD Students, all now either in academia (*e.g.,* University and CNRS) or industry.

Moreover, we devoted important efforts in transferring our research results, into education (*e.g.,* creation of specific supports for advanced courses), towards a more general audience (*e.g.* book publication [190]) and to industry (*e.g.,* creation and animation of the open-source research consortium GEMOC, together with the development of the Eclipse GEMOC Studio).

Finally, not only we continued our longstanding collaborations in the domain of systems engineering, but we also investigated new application domains to offer a broader impact to our research activities (cf. Section 2.4.4). In particular, we developed a new expertise in scientific computing, both for engineering (Safran, CEA) and environmental sciences (OSUR, Lancaster University).

2.5 Research Axis 2: Variability modeling

2.5.1 Personnel

Permanent fellows: M. Acher, J.-M. Jézéquel, Djamel E. Khelladi, A. Blouin, B. Combemale, O. Barais, B. Baudry
Postdocs: J. A. Galindo, X. Ternava, J. Pereira
PhD students: L. Lesoil, H. Martin, Q. Plazar, P. Temple

2.5.2 Project-Team Positioning

The systematic modeling of variability in software systems has emerged as an effective approach to document and reason about software evolution and heterogeneity (*cf.* Section B.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 tools for 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.

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 the 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 to ensure 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 an extremely large set of variants.

Model-driven development supports two approaches to deal with the increasing number of concerns in complex systems: 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 must 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.

Scientific objectives We aim at developing scalable reasoning techniques to **automatically analyze** variability models and their interactions with other views on the software intensive system (requirements, architecture, design, code). 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 and learning-based techniques for the exploration of the space of interaction between variability and view models.

We aim at developing 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.

2.5.3 Scientific Achievements

Modeling variability. We contribute to a set techniques and formalisms to specify models of variability. This axis mainly focuses on scenarios in which persons (domain experts, developers, testers, etc.) elaborate and write variability. This is in contrast with the other reverse engineering and learning axis in which variability is automatically obtained, refined or augmented.

The contribution "Modeling Variability in the Video Domain: Language and Experience Report" [19] reports on our experience in modelling variability in the video domain. We detail the design of the VM language with advanced constructs in such a way video experts can comprehensively express variability information. We show the lack of standard variability languages and basic feature models. It eventually leads to the development of the MOTIV industrial generator for synthesizing a diverse of realistic video variants in a controlled way.

In [132], we address the problem of exploring numerous architectural alternatives until choosing the most adequate variant. The work lies in a systems engineering context and is a collaboration with Thales. The decision-making process is most of the time a manual, time-consuming, and error-prone activity. The exploration and justification of architectural solutions is ad-hoc and mainly consists in a series of trial and error on the modeling assets. In this contribution, we report on an industrial case study in which we apply variability modeling techniques to automate the assessment and comparison of several candidate architectures (variants). We use and extend the Common Variability Language (CVL). Beyond variant selection automation improvement, these experiment results highlight that the approach improves rationality in the assessment and provides decision making arguments when selecting the preferred variants.

In [164], we introduce featured model types in order to transpose the product line paradigm at the metamodel level, where reusable assets are formed by metamodel and transformation fragments and "products" are reusable language building blocks (model types). The intent is to concisely model variability amongst metamodelling elements, enabling configuration, automated analysis, and derivation of tailored model types.

In [125], we present a formal analysis framework to analyze a family of platform products w.r.t. real-time properties. First, we propose an extension of the widely-used feature model, called Property Feature Model (PFM), that distinguishes features and properties explicitly. Second, we present formal behavioral models of components of a real-time scheduling unit such that all real-time scheduling units implied by a PFM are automatically composed to be analyzed against the properties given by the PFM. We apply our approach to the verification of the schedulability of a family of scheduling units using the symbolic and statistical model checkers of Uppaal.

Reverse engineering variability. We contribute to a set methods and techniques to reverse engineer models of variability. The goal is to create variability representations of artefacts in another form or at a higher level of abstraction. Many kinds of artefacts (languages, tabular data, etc.) are targeted.

In [60, 148, 149], we propose a reverse engineering technique to identify commonality and variation points out of language variants. Our approach receives a set of DSL variants that are used to automatically recover a language modular design and to synthesize the corresponding variability models. The validation is performed in a project involving industrial partners that required three different variants of a DSL for finite state machines. This validation shows that our approach is able to correctly identify commonalities and variability.

In [21], we show how to synthesize product comparison matrices (PCMs) out of informal product descriptions. Individual products are usually described in natural language (e.g., English) within textual data that are poorly structured and ambiguous. The challenge is to capture qualitative and quantitative information about the features of products and then synthesize a PCM. We develop natural language processing techniques to achieve this goal. We provide empirical evidence that there is a potential to complement or even refine technical information of products thanks to our extraction. The evaluation insights drive the design of the *MatrixMiner*, a web environment with an interactive support for synthesizing, visualising and editing PCMs.

Besides, we contributed to a large community involved in reverse engineering variability (see, e.g., the REVE workshops[3], co-organized with the SPLC conference).

Testing and sampling variability. Many approaches for testing configurable software systems start from the same assumption: it is impossible to test all configurations. This motivated the definition of variability-aware abstractions and sampling techniques to cope with large configuration spaces. Yet, there is no theoretical barrier that prevents the exhaustive testing of all configurations by simply enumerating them, if the effort required to do so remains acceptable. Not only this: we believe there is much to be learned by systematically and exhaustively testing a configurable system. In a case study [42, 111], we report on the first ever endeavour to test all possible configurations of an industry-strength, open source configurable software system, JHipster, a popular code generator for web applications. We built a testing scaffold for the 26,000+ configurations of JHipster using a cluster of 80 machines during 4 nights for a total of 4,376 hours (182 days) CPU time. We find that 35.70% configurations fail and we identify the feature interactions that cause the errors. We show that sampling strategies (like dissimilarity and 2-wise): (1) are more effective to find faults than the 12 default configurations used in the JHipster continuous integration; (2) can be too costly and exceed the available testing budget. We cross this quantitative analysis with the qualitative assessment of JHipster's lead developers.

This study shows that *sampling* techniques are of prior importance. The challenge is to generate valid configurations that respect constraints while meeting a certain criterion. Uniform random sampling is a strong baseline and is useful in many testing scenarios (but also learning scenarios, see hereafter). In *Uniform Sampling of SAT Solutions for Configurable Systems: Are We There Yet?* [166], we show that state-of-the-art solutions either do not scale or are not uniform. Much more research is needed to have scalable and effective sampling strategies, especially for large variability spaces.

The study *Sampling Effect on Performance Prediction of Configurable Systems: A Case Study* [83] (**Best paper award at ICPE 2019**) shows that there is no single "dominant" sampling strategy even for the same configurable system. Instead, different strategies perform best on different inputs and non-functional properties, further challenging practitioners and researchers.

In [67] we propose to apply so-called "Multimorphic testing" and empirically assess the effectiveness of performance test suites of software systems coming from various domains. By analogy with mutation testing, our core idea is to leverage the typical configurability of these systems, and to check whether it makes any difference in the outcome of the tests: i.e., are some tests able to "kill" underperforming system configurations? More precisely, we propose a framework for defining and evaluating the coverage of a test suite with respect to a quantitative property of interest. Such properties can be the execution time, the memory usage or the success rate in tasks performed by a software system. This framework can be used to assess whether a new test case is worth adding to a test suite or to select an optimal test suite with respect to a property of interest. We evaluate several aspects of our proposal through three empirical studies

[3] http://reveworkshop.github.io/

carried out in different fields: object tracking in videos, object recognition in images, and code generators.

Learning variability. We contribute to a set of methods and techniques based on supervised, statistical machine learning to model variability constraints and non-functional properties (performance) of a configurable system.

In [205, 145] we perform a systematic review of the rich literature related to the learning of software configuration spaces. We report on the different application objectives (e.g., performance prediction, configuration optimization, constraint mining), use cases, targeted software systems and application domains. We review the various strategies employed to gather a representative and cost-effective sample. We describe automated software techniques used to measure functional and non-functional properties of configurations. We classify machine learning algorithms and how they relate to the pursued application. Finally, we also describe how researchers evaluate the quality of the learning process.

The work *Using Machine Learning to Infer Constraints for Product Lines* [179] was the starting point of our journey. We describe a learning approach to find subtle constraints of an industrial video generator used in the MOTIV project. The motivation is that despite important engineering effort the MOTIV generator synthesizes sometimes bad video variants due to lack of constraints among parameters. Our solution samples the configuration space, trains with a decision tree, and then injects rules of the tree into the variability model. The seminal ideas were further developed in the context of Paul Temple's PhD thesis. In [66], we extend this line of work along two perspectives: (1) the learning of constraints can be parameterized by an high-level, performance objective; (2) constraints among the contextual features and the software configuration space can be automatically synthesized.

In [144] (**Best paper award at SPLC 2021**), we compare six learning techniques: three variants of decision trees (including a novel algorithm) with and without the use of model-based feature selection. We first perform a study on 8 configurable systems considered in previous related works and show that the accuracy reaches more than 90%, and that feature selection can improve the results in the majority of cases. We then perform a study on the Linux kernel and show that these techniques perform as well as on the other systems. Overall, our results show that there is no one-size-fits-all learning variant (though high accuracy can be achieved): we present guidelines and discuss tradeoffs.

In [68, 178] we introduce the use of adversarial machine learning to enforce a configuration classifier or pinpoint problematic cases. It allows to diagnose prediction errors and take appropriate actions. We develop two adversarial configuration generators on top of state-of-the-art attack algorithms; these generators are capable of synthesizing configurations that are both adversarial and conform to logical constraints. We empirically assess our generators with two case studies: an industrial video synthesizer (MOTIV) and an industry-strength, open-source Web-app configurator (JHipster). For the two cases, our attacks yield (up to) a 100% misclassification rate without sacrificing the logical validity of adversarial configurations. This work lays the foundations of a quality assurance framework for learning-based software product lines.

In [78], we develop the VaryLaTeX tool to synthesize papers' variants that meet constraints such as page limits. It is based on learning techniques to enforce constraints prior to the generation. In [84] we apply a learning-based process to mine constraints and prevent defects in the 3D printing domain.

In [203, 80, 58], we apply learning techniques to the Linux kernel. With now more than 15,000 configuration options, including more than 9,000 just for the x86 architecture, the Linux kernel is one of the most complex configurable open-source system ever developed. If all these options were binary and independent, that would indeed yield 2^{15000} possible variants of the kernel. Of course not all options are independent (leading to fewer possible variants), but some of them have tri-states values: yes, no, or module instead of simply boolean values (leading to more possible variants). The Linux kernel is mentioned in numerous papers about configurable systems and machine learning for motivating the problem and the underlying approach. However, only a few works truly explore such a huge configuration space. In this line of work, we take up the Linux challenge either for configurations' bug prevention or for predicting the binary size of a configured kernel. We also design a learning technique capable of transferring a prediction model among versions of Linux [58].

Teaching variability. We are co-leading a worldwide initiative[4] for disseminating the constantly growing body of software product line knowledge. This repository aims at sharing and delivering teaching material related to variability, configurable systems or generative approaches. This initiative is the result of three international workshops that we have co-organized, together with several surveys. An outcome of work is the article *Teaching Software Product Lines: A Snapshot of Current Practices and Challenges* [18], presented at SIGCSE'18.

2.5.4 Collaborations

During this period, we worked with KTH (testing variability), SnT (sampling variability), University of Namur (testing, sampling, and adversarial learning of variability), University of Delft (testing variability), University of Sevilla (modeling and learning variability), JKU Linz (teaching variability).

2.5.5 External support

The work was mainly supported by ANR JCJC VaryVary, MOTIV (DGA rapid), ANR Soprano, SLIMFAST (DGA and Britany region)

2.5.6 Self Assessment

Our publication record covers (1) the two main specialized events about variability: SPLC and VaMoS; (2) top-tier software engineering journals (TSE, EMSE, JSS). We won three awards (best paper awards at ICPE and SPLC, most influential paper at SLE). We are present in the steering committees of SPLC and VaMoS and have been program committee co-chairs of these events. We are organizing workshops or conducting tutorials on our research axis (teaching, reverse engineering, learning).

In the future, we plan to further submit to top-tier, general software engineering conferences though we feel more comfortable with top-tier journals. We also believe software variability can be applied in other computer science domains (e.g., security, artificial intelligence). Hence a future ambitious goal can be to broaden the scope of our contributions.

We are targeting important, well-established, complex, and active open-source software projects (JHipster, x264, Linux). Though our results are promising and at the state of art, we cannot claim we are at the point of integrating our results (e.g., prediction models) and having concrete impacts on developers and users. More research is definitely needed to understand the reasons of this situation and bridge the gap. Another important reason is that much more engineering support is needed to maturate, integrate, and follow the rapid continuous evolution of these projects.

[4] http://teaching.variability.io/

2.6 Research Axis 3: Runtime configuration and deployment for software-defined network services

2.6.1 Personnel

Permanent fellows: O. Barais, B. Baudry, J. Bourcier, J.-M. Jézéquel, N. Plouzeau.
Postdocs: A. Benenallam
PhD students: L. Mouline, A. Rio, J. Pelay, J-E Dartois, M. Boussaa, J. Bourgeois, A-A Pranata.

2.6.2 Project-Team Positioning

Flexible yet dependable systems have to cope with heterogeneous hardware execution platforms ranging from smart sensors to huge computation infrastructures and data centers. Evolution possibilities range from a simple 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 (e.g., 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 runtime. 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 unforeseen evolution cases.

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 reconfiguration* 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 reconfiguration.

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.

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, the number of testing environments to test a software patch or upgrade is the product of the number of execution environments the software supports and of 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 for 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 for and reserve resources on a distributed network of heterogeneous devices.

2.6.3 Scientific Achievements

Automatic synthesis of optimal software architectures.

Regarding this challenge, we keep working in the field of Models@runtime. The objective is to provide the appropriate abstraction at the modeling layer in order to simplify the reasoning and synthesis of the optimal software architecture. In this axis, one of the goals is to work on the shipping steps of the deployment phase, and the reconfiguration accessible to a whole development team. During the evaluation period, we worked in particular on the cloud computing domain and on the energy domain.

In the first one, we enhance existing models that could be used at runtime to improve resource usage in two domains: cloud computing [41, 110, 113, 103, 104, 38, 105, 86, 112, 155, 29, 22] and energy [213, 170, 169, 162].

In the cloud computing domain, we provide specific abstractions to cloud providers, in order to allow the reselling of unused resources to peers. Indeed, although cloud computing techniques have reduced the total cost of ownership thanks to virtualization, the average usage of resources (e.g., CPU, RAM, network, I/O) remains low. To address such issue, one may sell unused resources. Such a solution requires the cloud provider to determine the resources available and estimate their future use to provide availability guarantees. In this work, we propose a technique that uses machine learning algorithms (Random Forest, Gradient Boosting Decision Tree, and Long Short Term Memory) to forecast 24-hour of available resources at the host level. Our technique relies on the use of quantile regression to provide a flexible trade-off between the potential amount of resources to reclaim and the risk of SLA violations. In addition, several metrics (e.g., CPU, RAM, disk, network) are built to provide exhaustive availability guarantees. Our methodology is evaluated by relying on four in production data center traces, and our results show that quantile regression is relevant to reclaim unused resources. Our approach may increase the amount of savings up to 20% compared to traditional approaches [103, 104, 38, 105].

In the energy domain, we work at proposing models that could be used at runtime to improve self-consumption of renewable energies [213, 170, 169]. Self-consumption of renewable energies is defined as electricity that is produced from renewable energy sources, not injected to the distribution or transmission grid or instantaneously withdrawn from the grid and consumed by the owner of the power production unit or by associates directly contracted to the producer. Designing solutions in favor of self-consumption for small industries or city districts is challenging. It consists in designing an energy production system made of solar panels, wind turbines, batteries that fit the annual weather prediction and the industrial or human activity. In this context, we define a domain specific modeling language designed to let domain experts run their own simulations at design time and at runtime [213, 170, 169].

Flexible software architecture for testing and data management.

To answer this challenge, we have shown the benefit of disposing of a configurable software architecture in order to exhaustively test different configurations of the same system [198]. In particular, we have proposed an approach to generate dedicated architectures to test different code generators for heterogeneous platforms [94, 6, 25, 93]. This approach is based on the notion of meta-morphic testing in order to avoid the definition of an oracle by a tester. Using learning techniques our approach builds a model of resource usage for each target platform and then uses classification techniques to decide when a code generator for one of the target platforms is out of its validity envelop regarding resources consumption. In continuation of this work, we have shown how it is possible to use the same type of technique in the domain of virtualized functions in order to detect bugs during a reconfiguration [197, 167]. The proposed approach builds a nominal resource usage model for a given virtualized function using learning techniques, and it can thus detect an error during a reconfiguration (dependencies update, deployment topologies update, etc.) by detecting an execution outside the nominal operating envelop. As a complement, we also show how we could trigger an internal reconfiguration[5] in a testing environment in executing virtualized network functions under perturbation.

Runtime support for heterogeneous environments.

The Kevoree platform is the main result in which we have integrated the work done within this axis. This software platform provides an implementation in Java [153], in JavaScript [181] (in the browser or on the server), a support for the deployment of Docker components, a support for the deployment on embedded platforms such as Arduino micro-controllers. It offers a common abstraction layer allowing to describe in a declarative way the desired deployment. It has an execution platform and a development model for heterogeneous environments in order to interpret the desired configuration and control the local adaptations to be made. The reification through the notion of "group" of the coordination semantics of the adaptation allows to explain the behavior of each of these platforms. The Kevoree platform source amounts to 10,125 commits from 56 contributors, representing 1,065,549 lines of code.

During the evaluation period, we also extended the Models@runtime layer used within Kevoree. Indeed, the evolving complexity of adaptive systems impairs our ability to deliver anomaly-free solutions. Fixing these systems require a deep understanding on the reasons behind decisions that led to faulty or suboptimal system states. Developers thus need diagnosis support that trace system states to the previous circumstances –targeted requirements, input context– that had resulted in these decisions. However, the lack of efficient temporal representation limits the tracing ability of current approaches. To tackle this problem, we describe a novel temporal data model to represent, store and query decisions as well as their relationship with the knowledge (context, requirements, and actions). We validate our approach through a use case based on the smart grid at Luxembourg [157, 155, 156].

While developments around Kevoree have decreased somewhat over the past two years, it is worth noting that work on resource management has been actively pursued as part of the openstack open-source project through the collaboration with B-COM. The Watcher project[6] is an implementation of an autonomic loop clearly in the spirit of runtime models. The industrial dynamics in the field of deployment has led to the creation of large audience projects such as K8S, the work around the modeling of resources around a Models@runtime approach has been

[5] reconfiguration bound to an autonomic self-healing loop within the system

[6] https://wiki.openstack.org/wiki/Watcher

conducted in this context [113, 103, 104, 86, 112].

2.6.4 Collaborations

During the evaluation period, we actively collaborated internally at INRIA with the EASE project-team (2 joint PhDs), Myriads (1 joint PhD), Wide [100] (common contribution), Spirals (project, persons and platform transfer between the two groups and many common publications). Through our activities in the cloud computing domain, we regularly interact with the STACK team (PhD students follow-up, informal workshop, ...). During this period, we also worked actively with B-COM, KTH, SINTEF, and SnT. It should be noted that several PhD students who worked on the Kevoree platform have founded the DataThings company, which has been growing rapidly for the last two years in Luxembourg (https://datathings.com/). We keep exchanging regularly with this spinoff.

2.6.5 External support

- Occiware
- FP7 STREP HEADS
- ADR Nokia
- BCOM (CIFRE)
- Orange (CIFRE + CRE)
- OKWind (CIFRE)
- H2020 ICT-10-2016 STAMP

2.6.6 Self Assessment

The *Runtime configuration and deployment for software-defined network services* axis bridges the gap between the domains of software engineering, systems engineering and distributed systems. Our will to find the right abstractions to capture the behavior of applications and execution platforms remains a key motivation to propose the right development model and the right management model for modern applications. Proposing our own deployment platform has allowed us to be a precursor on the notion of Models@runtime. During the evaluation period, we continued this work in order to understand how to build resource models at the heart of modern systems and how to use these models to optimize the functioning of a software system.

Some highlights of our contribution and results are:

- the publications in well-establised conferences and journals (CCGRID, IEEE transactions on cloud computing, JSS, Models,...)
- the development of the Kevoree platform in the framework of a European project but also with several direct industrial partnerships
- The training of 7 PhD students in the field (all of them are now founders of innovative companies or tech leads in partner companies)

The ideas behind the notion of Models@runtime are now fully reflected in cloud orchestrators such as Kubernetes (K8S)[7]. In order to maximize the impact of our work, we have chosen to continue our work by integrating our current and future contributions within this platform. While the learning cost is huge and the granularity of the reconfiguration is higher compared to software reconfiguration primitives available in Kevoree, we benefit from a large spectrum of technical contributions from the community. During the evaluation period, we initiate the use of advanced deployment techniques for the security domain (through a partnership project called OneShot Software) and for the resilience and testing domain (see sub-axis 2). Further work on the problem of dynamic reconfiguration and deployment on heterogeneous platforms will be carried out mainly in the new #3 axis presented in the perspectives section.

[7] https://kubernetes.io/

2.7 Research Axis 4: Software implementation diversification

2.7.1 Personnel

Permanent fellows: O. Barais, B. Baudry, A. Blouin, B. Combemale, J.M. Jézéquel, D. Khelladi, N. Plouzeau, O. Zendra, Gurvan Le Guernic
Postdoc: N. Messe, A. Benenallam
PhDs: A. Hervieu, O. L. Vera-Pérez P. Temple, M. Boussaa, A. Gomès-Boix, P. Laperdrix, J. E. Dartois, Q. Le Dilavrec, Y. Ndiaye

Engineers: R. Lefeuvre, R. Bellafia

2.7.2 Project-Team Positioning

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 evolution cases 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 unforeseen situations. More specifically, we address the following V&V challenges.

Challenges One major challenge when building flexible and open yet dependable systems is that current software engineering techniques require architects to foresee all possible situations that the system will have to face. However, openness and flexibility also mean unpredictability: unpredictable bugs, attacks, environmental evolution, etc. Current fault-tolerance [Ran75] and security [FSA97] 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 works that rely on implementation 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, and to randomization of instruction sets. An essential and active 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.

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" [Rin12, ZMKR12].

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. 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 name multi-tier diversification the technique 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 logic 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 to 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.

2.7.3 Scientific Achievements

Proactive program diversification. On this axis, much work has been done in the field of proactive diversity of software and software tests in order to improve the quality of tests but also to evaluate the test suite quality [36, 72, 37, 43]. Thus, within the framework of the European project Stamp [8], we have highlighted the interest of *extreme mutation* in order to better qualify the quality of existing test suites [72, 70, 183]. We have also worked on the use of self-repair techniques to improve existing tests [24, 34]. We have worked extensively on the notion of patch generation at runtime [35, 49], the qualification of test data by automated variant synthesis in the context of extreme mutation []. We also propose a framework for defining and evaluating the coverage of a test suite with respect to a quantitative property of interest [67, 177].

Software evolution A second important aspect of automated diversification was to better understand the problems of co-evolution in software. Indeed, transforming a software artifact generally requires co-evolving associated artifacts. For example, transforming an abstract model requires co-evolving the generated code sometimes manually modified by the developer; transforming an API definition requires an evolution the client, which is sometimes developed using another programming language, etc. In this context, we started a project on the challenge of software co-evolution, thanks to the arrival of Djamel Eddine Khelladi in the team. With our first results we show how to co-evolve models and associated code [121, 44, 122, 50, 123] but also how to better understand the existing co-evolution in open source code repositories [128]. Another important work focused in detail on the natural evolutions found in binaries repositories in the Java community for example [89].

Multi-tier software diversification. This axis is the one that has probably evolved the most with respect to its initial objective. Several demonstrators have been built in the context of a partnership with Orange, aiming at demonstrating how to use adaptation techniques at the application architecture level in order to make several variants of the same application coexist.

[8] https://www.stamp-project.eu/

However, this study was conducted in the context of improving software security. While it was clear that software diversity improves the resilience of large systems, it was not clear how to provide evidence of its benefits for a single service. Thus, still in the context of the partnership with Orange, we worked on authentication in multi-tier applications and more generally on several aspects related to security with the arrival of Olivier Zendra in the DiverSE team.

Besides the Orange collaboration, we conducted studies on Maven artifacts that indicate that the immutability those artifacts does support a sustained level of diversity among versions of libraries in the repository [174]. We also conducted studies on browser fingerprint as an identification technique based on various information (cookies, hardware, software components) [182].

On the authentication axis, we have conducted studies on the openId Connect protocol and its use in the context of WebRTC session in order to show the difficulties of preserving the privacy of the user [32, 65, 33], we have worked on a language for the specification of authentication paths to support risk analysis associated with each authentification path [160]. We worked on the notion of browser finger printing both in a context of study of privacy [109, 107, 127, 126] but also in a context of its use for authentication [195]. This work had a notable impact in the community. Through the collaboration with the DGA, we have worked on the modeling of network packet filtering policies as well as their formalization in order to identify diversification possibilities [129, 207, 206, 130].

Finally, the arrival of Olivier Zendra allowed us to start looking at vulnerability and malware detection techniques in order to understand how they could be used in the context of advanced diversification issues [115, 151, 161, 168].

2.7.4 Collaborations

During the evaluation period, we actively collaborated internally at INRIA with the Spirals project team. We actively also work with Prof. Gildas Avoine at IRISA on browser finger printing. During this period, we also worked actively with B-COM, KTH, SINTEF. We also keep an active collaboration with Orange Labs and our partner [120, 44, 50, 49] at JKU, Austria.

2.7.5 External support

- Orange (3 * CIFRE + CRE)
- BCOM (CIFRE)
- H2020 ICT-10-2016 STAMP
- ANR JCJC x2

2.7.6 Self Assessment

This axis evolved over the period. This can be explained by significant human resources changes. With Benoit Baudry taking up his position at KTH, the arrival of Djamel Eddine Khelladi and Olivier Zendra, the two full-time researchers in the team, has allowed us to explore in greater detail the problems of co-evolution and to work more in the field of resilience and security. However, the academic results in this axis are excellent and well-published. Our work on test amplification is highly original while offering real industrial perspectives. Our work on browser finger printing had a real impact both at the academic and societal levels [9]. The theme is out of the scope of DiverSE for the next period, as two non-permanent DiverSE persons working on this topic have now permanent positions within the Spirals Project Team (Walter Rudametkin and Pierre Laperdrix). The long-standing collaboration with Orange on architecture and authentication challenges allows DiverSE to work regularly and in confidence with the Orange Labs's *authentication research team*. The work on co-evolution lead by Djamel Eddine Khelladi already provides excellent publications, one ANR JCJC project and has led to several ongoing PhDs in the team. Finally, the arrival of Olivier Zendra sharing the DiverSE scientific background but bringing his skills in the cybersecurity domain opens many perspectives for the next period.

[9] Prix INRIA/CNILL 2018

2.8 Evolution of Research Directions During the Evaluation Period

At the start of the current evaluation period, our planned research directions were:

- Language diversity: Software Language Engineering.
- Functional diversity: Variability modeling.
- Execution diversity: Runtime configuration and deployment of heterogeneous and distributed services.
- Implementation diversity: Software test amplification

Most of our research directions were kept during the evaluation period.

On the first two axes, we have deepened the scientific proposal, allowing the emergence of new sub-axes. In particular, the work on micro-servicisation of language services is promising and it has become one the challenges of the next period. The targeted application domains also had a real impact on the software platforms developed within the Gemoc initiative. The work on the notion of deep-variability is also at the core of the next period. On the *Execution diversity* axis, we have mainly worked on the problem of resource management. A large part of the work was carried out in the field of cloud computing. No new PhD thesis was started in the last three years. The last PhD students working on execution diversity will complete their work at the end of 2021 or beginning of 2022. This leads to a restructuring of this axis for the next evaluation period. On the *Implementation diversity* axis, three major events could be highlighted. The leave of Benoit Baudry is not visible in the 2016-2021 period. Indeed, several of Benoit's doctoral students defended their thesis during the period, and they remain within the team. However, as for axis #3, no PhDs in the field of software testing has started in the last three years. As a major topic evolution we have started to work on co-evolution thanks to the arrival of Djamel as a CR CNRS lead. Similarly, the arrival of Olivier Zendra in the team has opened up the field of software security.

Evolution of Research Directions During the Evaluation Period

The work program has been followed for the period 2016-2021. However, we would like to take advantage of this evaluation to refactor the team organization for the next evaluation period. Section 5 details this team reorganization.

3 Knowledge Dissemination

3.1 Publications

	number from October 2016 to October 2021
PhD Theses	18
HDRs (*)	2
Journal papers	55
Conference papers	109
Conference proceedings	3
Book chapters	3
Books	1
General audience papers	1
Technical reports	12

(*) HDR: Habilitation à diriger des Recherches

DiverSE's publications policy:

Over this period, we have voluntarily adopted a strategy of "journal first" publications without necessarily starting with an acceptance in a conference. This strategy, which is not exclusive either, is mainly motivated by the possibility offered by the journals to use the different rounds of submissions to explain certain ideas in breakthrough.

Publications in Main Journals:

1. IEEE Trans. on Software Engineering: 3
2. Sosym: 10
3. JSS: 7
4. Empirical Software Engineering: 6
5. STVR: 1
6. CACM: 1
7. IST: 3
8. COLA (formerly COMLAN): 4
9. Transactions on Modularity and Composition: 1
10. Transactions on Cloud Computing: 1

Publications in Major Conferences:

1. ICSE: 1
2. ASE: 1
3. MODELS: 5
4. SLE: 7
5. IEEE Symposium on Security and Privacy (SnP): 2
6. International World Wide Web Conference (WWW): 1
7. SPLC: 11
8. MSR: 3
9. CCgrid: 1
10. ICMSE: 1
11. ICST: 1
12. ICPE: 1
13. VRST: 1

14. ICSE/NIER: 1
15. Onwards: 2
16. GPCE: 1
17. IJCAI: 1
18. Privacy Enhancing Technology (CSP): 1

Selection of Publications The following papers will be available on the evaluation website:

1. **Co-Evolving Code with Evolving Metamodels**, in International Conference on Software Engineering (ICSE) 2020 [121]. This paper proposes a semi-automatic co-evolution code metamodel approach based on change propagation. The premise is that knowledge of the metamodel evolution changes can be propagated by means of resolutions to drive the code co-evolution. Our approach leverages on the abstraction level of metamodels where a given metamodel element has often different usages in the code. It supports alternative co-evaluations to meet different developers needs. Our work is evaluated on three Eclipse SL implementations, namely OCL, Modisco, and Papyrus over several evolved versions of metamodels and code. In response to five different evolved metamodels, we co-evolved 976 impacts over 18 projects.A comparison of our co-evolved code with the versioned ones shows the usefulness of our approach. Our approach was able to reach a weighted average of 87.4% and 88.9% respectively of precision and recall while supporting useful alternative co-evolution that developers have manually performed.
2. **Empirical Assessment of Multimorphic Testing** , IEEE Transactions on Software Engineering (TSE) 2019 [67]. This paper proposes to apply Multimorphic testing and empirically assess the effectiveness of performance test suites of software systems coming from various domains. By analogy with mutation testing, our core idea is to leverage the typical configurability of these systems, and to check whether it makes any difference in the outcome of the tests: i.e., are some tests able to "kill" underperforming system configurations? More precisely, we propose a framework for defining and evaluating the coverage of a test suite with respect to a quantitative property of interest. Such properties can be the execution time, the memory usage or the success rate in tasks performed by a software system.
3. **Omniscient Debugging for Executable DSLs** , Journal of Systems and Software (JSS) 2018 [26]. Omniscient debugging is a promising technique that relies on execution traces to enable free traversal of the states reached by a model (or program) during an execution. While a few General-Purpose Languages (GPLs) already have support for omniscient debugging, developing such a complex tool for any executable Domain Specific Language (DSL) remains a challenging and error-prone task. A generic solution must: support a wide range of executable DSLs independently of the metaprogramming approaches used for implementing their semantics; be efficient for good responsiveness. Our contribution relies on a generic omniscient debugger supported by efficient generic trace management facilities. To support a wide range of executable DSLs, the debugger provides a common set of debugging facilities, and is based on a pattern to define runtime services independently of metaprogramming approaches.
4. **Interacto: A Modern User Interaction Processing Model** , IEEE Transactions on Software Engineering (TSE) 2021 [23]. In this paper, we propose Interacto as a high-level user interaction processing model. By reifying the concept of user interaction, Interacto makes it easy to design, implement and test modular and reusable advanced user interactions, and to connect them to commands with built-in undo/redo support. To demonstrate its applicability and generality, we briefly present two opensource implementations of Interacto for Java/JavaFX and TypeScript/Angular.
5. **Investigating Machine Learning Algorithms for Modeling SSD I/O Performance for Container-based Virtualization**, IEEE Transactions on Cloud Computing 2020 [38]. We work on a framework based on autonomic computing, which aims to achieve

intelligent container placement on storage systems by preventing bad I/O interference scenarios. One prerequisite to such a framework is to design SSD performance models that take into account interactions between running processes/containers, the operating system and the SSD. These interactions are complex. In this paper, we investigate the use of machine learning for building such models in a container-based Cloud environment. We have investigated five popular machine learning algorithms along with six different I/O intensive applications and benchmarks. We analyzed the prediction accuracy, the learning curve, the feature importance and the training time of the tested algorithms on four different SSD models. Beyond describing modelling component of our framework, this paper aims to provide insights for cloud providers to implement SLO compliant container placement algorithms on SSDs. Our machine learning-based framework succeeded in modeling I/O interference with a median Normalized Root-Mean-Square Error (NRMSE) of 2.5%.

3.2 Software

3.2.1 FAMILIAR

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.

Functional Description: Familiar is an environment for large-scale product customisation. From a model of product features (options, parameters, etc.), Familiar can automatically generate several million variants. These variants can take many forms: software, a graphical interface, a video sequence or even a manufactured product (3D printing). Familiar is particularly well suited for developing web configurators (for ordering customised products online), for providing online comparison tools and also for engineering any family of embedded or software-based products.

URL: http://familiar-project.github.com

Contact: Mathieu Acher

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

Partners: CNRS, University Nice Cote d'Azur

INRIA taxonomy: research, partners, lts, leader.

Duration of the Development More than 10 years

3.2.2 GEMOC Studio

Name: GEMOC Studio

Keywords: DSL, Language workbench, Model debugging

Scientific Description: The language workbench put together the following tools seamlessly integrated to the Eclipse Modeling Framework (EMF):

- Melange, a tool-supported meta-language to modularly define executable modeling languages with execution functions and data, and to extend (EMF-based) existing modeling languages.
- MoCCML, a tool-supported meta-language dedicated to the specification of a Model of Concurrency and Communication (MoCC) and its mapping to a specific abstract syntax and associated execution functions of a modeling language.

- GEL, a tool-supported meta-language dedicated to the specification of the protocol between the execution functions and the MoCC to support the feedback of the data as well as the callback of other expected execution functions.
- BCOoL, a tool-supported meta-language dedicated to the specification of language coordination patterns to automatically coordinates the execution of, possibly heterogeneous, models.
- Sirius Animator, an extension to the model editor designer Sirius to create graphical animators for executable modeling languages.

Functional Description: The GEMOC Studio is an Eclipse package that contains components supporting the GEMOC methodology for building and composing executable Domain-Specific Modeling Languages (DSMLs). It includes two workbenches: The GEMOC Language Workbench: intended to be used by language designers (aka domain experts), it allows to build and compose new executable DSMLs. The GEMOC Modeling Workbench: intended to be used by domain designers to create, execute and coordinate models conforming to executable DSMLs. The different concerns of a DSML, as defined with the tools of the language workbench, are automatically deployed into the modeling workbench. They parametrize a generic execution framework that provides various generic services such as graphical animation, debugging tools, trace and event managers, timeline.

URL: `http://gemoc.org/studio.html`

Authors: Didier Vojtisek, Benoît Combemale, Cédric Brun, François Tanguy, Joël Champeau, Julien DeAntoni, Xavier Crégut

Contacts: Benoît Combemale, Julien DeAntoni

Participants: Didier Vojtisek, Dorian Leroy, Erwan Bousse, Fabien Coulon, Julien DeAntoni

Partners: IRIT, ENSTA, I3S, OBEO, Thales TRT

INRIA taxonomy: research, community, lts, leader.

Duration of the Development More than 6 years

3.2.3 ALE

Name: Action Language for Ecore

Keywords: Meta-modeling, Executable DSML

Functional Description: Main features of ALE include:

- Executable metamodeling: Re-open existing EClasses to insert new methods with their implementations
- Metamodel extension: The very same mechanism can be used to extend existing Ecore metamodels and insert new features (eg. attributes) in a non-intrusive way
- Interpreted: No need to deploy Eclipse plugins, just run the behavior on a model directly in your modeling environment
- Extensible: If ALE doesn't fit your needs, register Java classes as services and invoke them inside your implementations of EOperations.

URL: `http://gemoc.org/ale-lang/`

Contact: Benoît Combemale

Partner: OBEO

INRIA taxonomy: research, partners, basic.

Duration of the Development More than 5 years

3.2.4 InspectorGuidget

Keywords: Static analysis, Software testing, User Interfaces

Functional Description: InspectorGuidget is a static code analysing tool. InspectorGuidget analyses UI (user interface/interaction) code of a software system to extract high level information and metrics. InspectorGuidget also finds bad UI coding pratices, such as Blob listener instances. InspectorGuidget analyses Java code.

URL: https://github.com/diverse-project/InspectorGuidget
Publications: hal-01499106v5, hal-01308625v2
Contact: Arnaud Blouin
Participants: Arnaud Blouin, Benoit Baudry
INRIA taxonomy: research, partners, basic, leader.
Duration of the Development More than 5 years

3.2.5 Descartes

Keywords: Software testing, Mutation analysis
Functional Description: Descartes evaluates the capability of your test suite to detect bugs using extreme mutation testing.
Descartes is a mutation engine plugin for PIT which implements extreme mutation operators as proposed in the paper *Will my tests tell me if I break this code?*.
URL: https://github.com/STAMP-project/pitest-descartes
Publications: hal-01870976, hal-01867423
Contacts: Benoit Baudry, Oscar Luis Vera Perez, Olivier Barais, Martin Monperrus
Participants: Oscar Luis Vera Perez, Benjamin Danglot, Benoit Baudry, Martin Monperrus
Partner: KTH Royal Institute of Technology
INRIA taxonomy: research, community, basic.
Duration of the Development More than 3 years

3.2.6 PitMP

Name: PIT for Multi-module Project
Keywords: Mutation analysis, Mutation testing, Java, JUnit, Maven
Functional Description: PIT and Descartes are mutation testing systems for Java applications, which allows you to verify if your test suites can detect possible bugs, and so to evaluate the quality of your test suites. They evaluate the capability of your test suite to detect bugs using mutation testing (PIT) or extreme mutation testing (Descartes). Mutation testing does it by introducing small changes or faults into the original program. These modified versions are called mutants. A good test suite should able to kill or detect a mutant. Traditional mutation testing works at the instruction level, e.g., replacing "¿" by "¡=", so the number of generated mutants is huge, as the time required to check the entire test suite. That's why Extreme Mutation strategy appeared. In Extreme Mutation testing, the whole body of a method under test is removed. Descartes is a mutation engine plugin for PIT which implements extreme mutation operators. Both provide reports combining, line coverage, mutation score and list of weaknesses in the source.
URL: https://github.com/STAMP-project/pitmp-maven-plugin
Contact: Caroline Landry
Partners: CSQE, KTH Royal Institute of Technology, ENGINEERING
INRIA taxonomy: research, community, basic, leader.
Duration of the Development More than 3 years

3.2.7 Kevoree

Keywords: M2M, Dynamic components, Iot, Heterogeneity, Smart home, Cloud, Software architecture, Dynamic deployment
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/doc/1.4/frascati-userguide.html).
- 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).

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.

URL: http://kevoree.org/

Authors: Jean Emile Dartois, Aymeric Hervieu, Olivier Barais

Contact: Olivier Barais

Participants: Aymeric Hervieu, Benoit Baudry, Francisco-Javier Acosta Padilla, Inti Gonzalez Herrera, Ivan Paez Anaya, Jacky Bourgeois, Jean Emile Dartois, Johann Bourcier, Manuel Leduc, Maxime Tricoire, Mohamed Boussaa, Noël Plouzeau, Olivier Barais

INRIA taxonomy: research, partners, no future, leader.

Duration of the Development More than 5 years

3.2.8 amiunique

Name: amiunique

Keywords: Privacy, Browser fingerprinting

Scientific Description: The amiunique web site has been deployed in 2014 in the context of the DiverSE team research activities on browser fingerprinting to understand how software diversity can be leveraged to mitigate the impact of fingerprinting on the privacy of users. In 2018, it was migrated to the Spirals team where the research on browser fingerprinting still continues to this day.

The website has yielded multiple datasets of genuine fingerprints to understand the multiple facets of browser fingerprinting and how they can be used on the web to reinforce security. The website presents regular updates to include the latest development in web technology and understand their impact of users' privacy.

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

Main innovative features:

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

Impact: The website has been visited by more than 3,000,000 unique visitors since its creation and it has been showcased in several professional forums and tutorial sessions over the years. It produced multiple datasets over the years that were used in articles published in top-tier conferences. Amiunique has received in 2018 the prize "Protection de la vie privée" granted by Inria and the CNIL. The research around fingerprints in amiunique has also been a source of influence for the Brave web browser.

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).

URL: https://amiunique.org/

Authors: Pierre Laperdrix, Antonin Durey, Walter Rudametkin Ivey

Contacts: Benoit Baudry, Pierre Laperdrix

Partners: INSA Rennes, Université de Lille

INRIA taxonomy: research, universe, lts, instigator (previous leader and now managed by Spirals team).

Duration of the Development More than 3 years

3.2.9 Melange

Name: Melange

Keywords: Model-driven engineering, Meta model, MDE, DSL, Model-driven software engineering, Dedicated langage, Language workbench, Meta-modelisation, Modeling language, Meta-modeling

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-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 DSLs. 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 tightly integrated with the Eclipse Modeling Framework ecosystem and relies on the meta-language Ecore for the definition of the abstract syntax of DSLs. Executable meta-modeling is supported by weaving operational semantics defined with Xtend. Designers can thus easily design an interpreter for their DSL in a non-intrusive way. Melange is bundled as a set of Eclipse plug-ins.

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.

URL: http://melange-lang.org

Contacts: Benoît Combemale, Thomas Degueule, Olivier Barais, Jean-Marc Jézéquel, Didier Vojtisek

Participants: Arnaud Blouin, Benoît Combemale, David Mendez Acuna, Didier Vojtisek, Dorian Leroy, Erwan Bousse, Fabien Coulon, Jean-Marc Jézéquel, Olivier Barais, Thomas Degueule

INRIA taxonomy: research, partners, basic, leader.

Duration of the Development More than 5 years

3.2.10 DSpot

Keywords: Software testing, Test amplification

Functional Description: DSpot is a tool that generates missing assertions in JUnit tests. DSpot takes as input a Java project with an existing test suite. As output, DSpot outputs new test cases on console. DSpot supports Java projects built with Maven and Gradle

URL: https://github.com/STAMP-project/dspot

Authors: Simon Allier, Benoit Baudry, Marcelino Rodriguez Cancio, Martin Monperrus

Contacts: Benoit Baudry, Benjamin Danglot

Participants: Benoit Baudry, Martin Monperrus, Benjamin Danglot

Partner: KTH Royal Institute of Technology

INRIA taxonomy: research, partners, basic, instigator.

Duration of the Development More than 3 years

3.2.11 Overall evaluation

GEMOC (which includes Melange and ALE) and Familiar will remain two core platforms of the team for the next evaluation period. We have aggregated and we keep aggregating the results of several PhD students into these two platforms. The maintenance and the evolution of AmIUnique are now taken in charge by the INRIA Spirals. Kevoree has brought many contributions including some libraries developed in the framework of Kecoree which are still used massively by other open source projects (*npmi* is still downloaded more than **20000 times** a week https://www.npmjs.com/package/npmi). However, on the deployment axis, we now come to place ourselves in the core of open-source project such as K8S to bring our idea to existing communities. The platforms developed within the framework of the STAMP project have given rise to numerous experiments by industrialists. Descartes is still at the heart of the continuous integration chain of various industrial players.

3.3 Technology Transfer and Socio-Economic Impact

During the period, members of DiverSE had close collaborations with

- large companies like Safran, Nokia Bell Labs, Orange, Thales
- small and medium-sized enterprises (SME) like XWiki, Fabernovel, OKwind, Obéo,
- and Institute like DGA (Rennes), CEA (Saclay & Bruyères le chatel) or IRT B-COM.

The Heads FP7 project has led to a transfer to TellU SaS of ThingML in which we provide several valuable contributions. ALE is now supported by Obeo.

Most of our former PhD students or PostDocs are now R&D engineers in companies or startups such as Amazon, Mathworks, Orange, XWiki, MediaKind, Famoco, OKWind, SHARE NOW, Adobe, KUGU, ActiveEon, SopraSeria, EXFO or Couture Technologies.

We devoted leadership within the Open Source Foundation Eclipse. In particular, we initiated the only one research consortium hosted within the Eclipse Foundation (the GEMOC Research Consortium). It is a working group that brings together academia and industry on the topic of software language engineering.

Finally, we currently initiate a startup project on the basis of our work on language engineering in the context of systems engineering. The project is matured since early 2021, and we aim at starting the company early 2022.

3.4 Teaching

The DiverSE team bears the bulk of the teaching on Software Engineering at the University of Rennes 1 and at INSA Rennes, for the first year of the Master of Computer Science (Project Management, Object-Oriented Analysis and Design with UML, Design Patterns, Component Architectures and Frameworks, Validation & Verification, Human-Computer Interaction) and for the second year of the MSc in software engineering (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, Benoit Combemale, Johann Bourcier, Arnaud Blouin, Stéphanie Challita and Mathieu Acher teaches about 250h in these domains for a grand total of about 2000 hours, including several courses at IMT-Atlantique, ENS Rennes and ENSAI Rennes engineering school[10].

The DiverSE team also hosts several MSc and summer trainees every year.

Responsibilities:

- Olivier Barais is **Deputy Director** of the electronics and computer science teaching department of the University of Rennes 1 (1600 students)
- Olivier Barais is the head of the final year of the Master in Computer Science at the University of Rennes 1
- Noel Plouzeau was the head of the first year of the Master in Computer Science at the University of Rennes 1 since september 2021
- Mathieu Acher will be the head of the first year of the Master in Computer Science at the University of Rennes 1 until August 2025
- Johann Bourcier is head of the Computer Science Department at the ESIR engineering school in Rennes
- Johann Bourcier is co-manager of the Home-Automation option at the ESIR engineering school in Rennes
- Arnaud Blouin is in charge of industrial relationships for the computer science department at INSA Rennes
- Stéphanie Challita is member of the Sustainable Development & Social Responsibility unit at the ESIR engineering school in Rennes

Commitment in Summer Schools:

- Mathieu Acher gave courses to EJCP (Ecole Jeune Chercheur en Programmation), a well-known, national training school for PhD students `https://ejcp2018.sciencesconf.org/resource/page/id/5` (2018, 2019, 2021)

3.5 General Audience Actions

- Jean-Marc Jézéquel has written several articles or had given interviews for Interstice, Science Ouest and the blog “Binaire” of “Le Monde”.

[10]All Professors or Associate Professors have a teaching duty of 192 hours per year, except B. Combemale. (96h) currently on leave (half time) at CNRS, and J-M Jézéquel (96h) when he was dean of IRISA. Djamel and Olivier, as Full-time researchers, have no mandatory teaching duties.

- STAMP - Orange Test and Dev Day, Grenoble, France - Nov 2019
- STAMP - Eastern Paris Dev Meetup, Lunatech, Chessy, France - Jul 2019
- STAMP Breakout Session & Talks - OW2con'19, Orange Gardens, Paris Châtillon - Jun 2019
- STAMP Workshop for EC-DGIT (European Commission DGIT) - Brussels - May 2019
- STAMP - Devoxx Paris - Apr 2019
- STAMP & CI testing automation - BreizhCamp 2019 - Mar 2019
- STAMP Talk and Workshop - Station-F, Paris - Feb 2019
- Tech talk at Alten - Rennes - Dec 2019
- Tech talk at Harmonic - Cesson-Sévigné - Oct 2019
- Tech talk at Veonum - Rennes - Sep 2019
- Tech talk at Nokia - Rennes - Jun 2019
- Tech talk at OrangeLabs : Hands-on Workshop - Cesson-Sévigné - May 2019
- Tech talk at Solocal and Kereval - Rennes - Mar 2019
- Tech talk at FaberNovel - Test amplification - Mar 2019
- Sciences de l'Ingénieur au féminin - Lycée Sévigné, Cesson-Sévigné - Nov 2019
- *J'peux pas j'ai informatique* - INRIA, Rennes - Apr 2019
- *Petit-déjeuenr des métiers* - Lycée Sévigné, Cesson-Sévigné - Mar 2019 (je ne suis pas sure de la date)
- Devoxx 2019 - Les APIs hypermedia expliquées simplement (17-19 avril 2019) ("While web pages are full of hyperlinks and hypermedia, web APIs are not. However, augmenting web APIs with links is at the core of the REST architectural style, but only 5% of APIs actually do it. That's right, only 5%, despite most people saying that they provide RESTful APIs. In this talk, we discuss what an hypermedia-powered API looks like, what problems of distributed systems integration and composition it can solve and what it would change in frontend code to have such APIs.
- Embedded Linux conference Europe [80] – Given a configuration, can humans know in advance the size, the compilation time, or the boot time of a Linux kernel? Owing to the huge complexity of Linux (there are more than 15000 options with hard constraints and subtle interactions), machines should rather assist contributors and integrators in mastering the configuration space of the kernel. In this talk, we introduce TuxML an OSS tool based on Docker/Python to massively gather data about thousands of kernel configurations. Mathieu will describe how 200K+ configurations have been automatically built and how machine learning can exploit this information to predict properties of unseen Linux configurations, with different use cases (identification of influential/buggy options, finding of small kernels, etc.) The vision is that a continuous understanding of the configuration space is undoubtedly beneficial for the Linux community, yet several technical challenges remain in terms of infrastructure and automation.
- API Days - Backend is the new frontend (9-11 décembre 2019)

3.6 Visibility

3.6.1 Awards

- 2021: [144] Best paper award at SPLC 2021
- 2021: [168] Best paper award at ARES-IWCC 2021
- 2021: [144] Best paper award at SPLC 2021
- 2021: Mathieu Acher Junior member **IUF** [11]
- 2020: Jean-Marc Jézéquel is the recipient of the IEEE/ACM International Conference on Model Driven Engineering Languages and Systems (MODELS) career award. [12]
- 2020: [83] Best paper award at ICPE 2020

[11] https://www.iufrance.fr/detail-de-lactualite/247.html
[12] https://shorturl.at/lFQV0

- 2019: We received the "Data Showcase Award" at the MSR'19 conference (Mining Software Repositories 2019) for the dataset described in the following paper [89] and publicly available on Zenodo[13].
- 2019: Most Influential Paper (MIP) award at SLE 2019
- 2018: [101] gets the best vision paper at SLE'18.
- 2018: [134] get the best artefact associated to a scientific paper at SLE'18.
- 2018: Winners of the third CNIL-Inria prize for their article [127] "*Beauty and the Beast: diverting modern web browsers to build unique browser fingerprints*", Benoît Baudry, Pierre Laperdrix and Walter Rudametkin.
- 2017: Second position for the ACM Student Research Competition: Thomas Degueule
- 2016: Silver Medal of the CNRS for Jean-Marc Jézéquel

3.6.2 Tutorials and Invited Conferences

Mathieu Acher:

- *Reverse Engineering Language Product Lines from Existing DSL Variants*, David Méndez-Acuña, José Galindo, Benoit Combemale, Arnaud Blouin, Benoit Baudry, at GDR-GPL 2018
- *Re-engineering Software Variability into Software Product Lines*, Tewfik Ziadi, Mathieu Acher, at ASE 2018 (tutorial: `http://www.ase2018.com/?p=tutorials#spl`)
- Tutorials at SPLC 2019 and 2020 about machine learning and SPLs
- Talk at Embedded Linux Conference Europe 2019 `https://www.youtube.com/watch?v=UBghs-cwQX4&feature=youtu.be`

Jean-Marc Jézéquel:

- *Keynote speech at ECMFA 2021*
- *Keynote speech for Performance Modeling workshop associated to ICPE2021*
- *Keynote speech Models@Runtime workshop associated to MODELS2019*
- *Keynote speech for the LIG seminar on the Future of Informatics: "On Turning Domain Knowledge into Tools"*

Djamel E. Khelladi:

- Talk at the *IPA Fall Days* in Wageningen, Netherlands, 2019.
- Invited talk at the GEODES team, University of Montreal, 2019.

Johann Bourcier:

- Talk at the UBS university in Vannes, France.

Benoit Combemale:

- *Smart Modeling: On the Convergence of Scientific and Engineering Models*, Invited talk at Lancaster university, UK (14/11/19).
- *Bringing Intelligence to Sociotechnical IoT Systems: Modeling Opportunities and Challenges*, Keynote at MDDE4IoT'19, DE (15/09/19).
- *Breathe Life Into Your IDE*, Talk at LangDev'19, NL (22/03/19).
- Model Execution: Past, Present and Future. Keynote at EXE'18, DK (14/10/18).
- Modeling For Sustainability - Or How to Make Smart CPS Smarter?. Keynote at Models@run.time'18, DK (14/10/18), and invited talk at TU Wien, AT (15/05/18)
- Execution Framework of the GEMOC Studio. Talk at LangDev'18, NL (08/03/18).
- From Model (driven) Engineering, to Language (driven) Engineering, Invited talk at CEA DAM,France.
- Towards Language-Oriented Modeling, Invited talk at UQAM (Montreal, Canada).
- Sound, yet Flexible, Modeling: A Language Engineering Point Of View, Keynote at FlexMDE'17.
- Modeling For Sustainability – Or How to Make Smart CPS Smarter? Invited talk at CWI (Amsterdam, The Netherlands).

[13] `https://zenodo.org/record/1489120`

- Model Simulation, Graphical Animation, and Omniscient Debugging with Sirius Animator. Invited talk at SiriusCon'17.

Benoit Baudry:

- Software technology and DevOps.
- Scientific days of Orange
- Reconciling Diversity and Privacy at the Dagstuhl Seminar on Online Privacy and Web Transparency
- Software diversification as an obfuscation technique, at the International Workshop on Obfuscation:Science, Technology, and Theory

Stéphanie Challita:

- Talk at ENS Rennes, France.

3.6.3 Program Committees, Editorial Boards, Conference Organization and Research Expertise

General Chair, Scientific Chair

General Chair

We organized ICPE 2021. Johann Bourcier served as general chair.
We organized MODELS 2016. Benoit Combemale and Benoit Baudry served as general chairs.

- Benoit Combemale was the PC co-chair for ICT4S 2020
- Benoit Combemale was the Tools & Demos co-chair for MODELS'20
- Benoit Combemale was the PC chair (Foundation track) for ECMFA 2019, ICMT'19
- Mathieu Acher was the program committee co-chair of the 14th International Working Conference on Variability Modelling of Software-Intensive Systems (VaMoS 2020)
- Mathieu Acher was the program committee co-chair of the SPLC 2017
- Mathieu Acher organized the Fourth international workshop on software product line teaching (SPLTea 2019).
- Mathieu Acher organized the First international workshop on languages for modelling variability (MODEVAR 2019).
- Mathieu Acher organized the Seventh international workshop on reverse variability engineering (REVE 2019).
- Mathieu Acher organized the 3rd ACM SIGSOFT International Workshop on Machine Learning Techniques for Software Quality Evaluation, MaLTeSQuE@ESEC/ SIGSOFT FSE 2019.
- Benoit Combemale has been appointed in 2018 Deputy Editor in Chief of the international journal JOT, and chair of the steering committee of the ACM SIGPLAN conference SLE.
- Benoit Combemale has been co-chair for the Industry Day at MODELS 2018, and the GEMOC Workshop at MODELS 2018.
- Benoit Combemale has been general chair for SLE 2017, the major international conference in the area of software language engineering.
- Benoit Combemale has been main organizer of the Dagstuhl Seminar #17342 on "The Software Language Engineering Body of Knowledge" (SLEBoK).
- Mathieu Acher has been program committee co-chair of SPLC 2017, the major international conference in the area of variability and software product line engineering.

Member of the Organizing Committees

- 2021: Benoit Combemale was the Finance Chair for ICPE'21
- 2021: Stéphanie Challita was Publication chair, ICSA'21
- 2021: Olivier Barais was is the organizing Committee of "*Les journées Clouds*".

- 2021: Djamel Khelladi was the Doctoral track co-chair at ICSSP/ICGSE 2021, collocated with ICSE
- 2020: Djamel Khelladi was the Poster co-chair at MODELS 2020
- 2020: Arnaud Blouin is member of the organising committee of JDev'2020
- 2019: Olivier Barais was the financial chair of the IEEE Mascots conference.
- 2018: Benoit Combemale has been a co-organizer of the session "Application of Advanced Software Engineering Tools and Methods in the Environmental Sciences" at CMWR'18.
- 2018: Mathieu Acher was Publicity chair, ICSR'18
- 2017: Arnaud Blouin was Publicity co-chair, EICS'17, 2017
- 2017: Mathieu Acher co-organized REVE'17 (international workshop on reverse engineering variability

Technical Program Committee (TPC)

Jean-Marc Jézéquel. PC member for SEAMS 2021 , SPLC 2021 Industry Track, ICSE 2020, SPLC 2020 , SEAMS 2020, SPLC 2019 Industry Track, SEAMS 2019, SPLC 2018 Industry Track, SEAMS 2018, ICSE 2017, SPLC 2017 Vision track, SEAMS 2017.

Olivier Barais. PC member for MODELS 2021, Compass 2020, CCgrid 2020, ICSR 2020, SecureMDE 2020 workshop at MODELS'20, the 17th International Conference on Software Engineering and Formal Methods (SEFM 2019), CIEL 2019, VaMoS 2019, VoSE 2019, SAC 2019, SATTA track - Software Architecture: Theory Technology and Applications, ICWE 2018. (Full Research Papers), ICWE 2018 (Short and Vision Papers), SAC'2018, SecureMDE 2018 workshop at MODELS'18, PC member SEAA 2017, PC member SAC 2017, PC member ICWE 2017.

Djamel E. Khelladi. PC member for MSR 2021, FSE artifact track 2021, IWOR Workshop 2021, collocated with ASE, ModDit Workshop 2021, collocated with MODELS, Models and Evolution (ME) Workshop 2021, collocated with MODELS, WETICE 2021 VSC Track, Models and Evolution (ME) Workshop 2020, collocated with MODELS, WETICE 2020 VSC Track, WETICE 2019 VSC Track, Poster track at MODELS 2019 , REVE workshop 2019, Models and Evolution (ME) Workshop 2019, collocated with MODELS, WETICE 2019 Validating Software for Critical Systems Track.

Arnaud Blouin. PC member for SAC'2020, PACM EICS Q4 2019, Workshop on Human Factors in Modeling at MODELS'2019 (HuFaMo), 2019, 30th French speaking conference on Human-Computer Interaction (IHM) 2018, Second International Workshop on Debugging in Model-Driven Engineering at MODELS'2018 (MDEBug), Third International Workshop on Human Factors in Modeling at MODELS'2018 (HuFaMo), the ACM Student Research Competition (SRC) at MODELS 2018, the ACM Student Research Competition (SRC) at MODELS 2017.

Mathieu Acher: PC member for SPLC 2016 - 2017- 2018 - 2019 - 2020 - 2021 Research Track, ICSE SEIP 2020, VaMoS 2016 - 2017 - 2018 - 2019 - 2020, 33rd IEEE/ACM International Conference on Automated Software Engineering (ASE), ICSR 2017, VaMoS 2017, SAC 2017, SEAA 2017, GramSec'17

Benoit Combemale:

- Program board member for MODELS'21, RE'21, ISEC'21, ECMFA'21, EASE'21, EduSymp at MODELS'21, DevOps4CPS-Testing at ICST'21, the Eclipse SAAM Mobility 2021 conference, MODELS'20, ECMFA'20, FDL'20, CBI'20, QUATIC'20, the MLE'20 workshop at MODELS'20, the DevOps@MODELS'20 workshop at MODELS'20, the CoMoDiTy'20 workshop at ER'20, the PNSE'20 workshop, the MoSC'20 workshop, MODELS'18,

ICMT'18, ECMFA'18, the MiSE'18 workshop at ICSE'18, the EXE'18 workshop at MODELS'18
- Selection Committee ACM SRC for MODELS'20 and the Doctoral Symposium at MODELS'19
- PC member for the MLE'19 workshop at MODELS'19, the DevOps@MODELS'19 workshop at MODELS'19, the FlexMDE'19 workshop at MODELS'19, the PNSE'19 workshop, the MDEBug'17 workshop at MODELS'18, the DevOps'18 workshop, MODELS'17, ICMT'17, the GEMOC'17 workshop at MODELS'17, the MDEbug'17 workshop at MODELS'17, the EXE'17 workshop at MODELS'17, the MiSE'17 workshop at ICSE'17, the MoMo'17 workshop at Modularity'17

Benoit Baudry. PC Member for ICSE 2017, ASE 2017

Johann Bourcier. PC Member for the SEsCPS'17 workshop at ICSE, the eCAS'18, eCAS'19, eCAS'20, eCAS'21 workshop at SASO, ICPE 2021 conference

Olivier Zendra. PC member for the ICOOOLPS'21 workshop at ECOOP'21

Stéphanie Challita. PC member for SLE 2021 Artifact Evaluation Track, MODELS 2021 Tools & Demonstrations Track, Models@runtime workshop at MODELS 2021, MLE workshop at MODELS 2021, FAACS workshop at ECSA 2021, ICSA 2021 Artifact Track, ICSA 2021 ECR Track, FormaliSE workshop at ICSE 2021, SAC 2021

Editorial Boards:
Jean-Marc Jézéquel:
- **Associate Editor-in- Chief** of SOSYM
- **Associate Editor-in-Chief** of IEEE Computer
- Associate Editor of JSS
- Associate Editor of JOT

Benoit Combemale:
- **Deputy Editor-in-Chief** of JOT
- Member of the Editorial Board of the Springer Journal on Software and Systems Modeling (SoSyM)
- Member of the Editorial Board of the Elsevier Journal of Computer Languages (COLA)
- Member of the Editorial Board of the Springer Journal of Software Quality
- Member of the Advisory Board of the Elsevier Journal on Science of Computer Programming (SCP, Software Section)

Benoit Baudry:
- SOSYM
- STVR

Olivier Zendra:
- Member of the Editorial Board of the HiPEAC Vision.

3.6.4 Leadership within the Scientific Community

Benoit Baudry:
- Steering committee member for the ACM/IEEE MODELS conference

Jean-Marc Jézéquel:
- Vice President of *InformaticsEurope*
- Member of the Executive Committee of GDR GPL
- Member of the Advisory Board of the GEMOC Initiative

Benoit Combemale:

- Chair of the Steering committee of the ACM SIGPLAN conference SLE
- Founding member and member of the advisory board of the GEMOC initiative.
- Chair of the Eclipse Research Consortium GEMOC and the Eclipse Project GEMOC Studio.
- Founding member of MLE and ModDit workshops

Mathieu Acher:
- Member of the Steering committee of SPLC conference
- Member of the Steering committee of VaMoS conference

Arnaud Blouin:
- founding member and co-organiser of the French GDR-GPL research action on Software Engineering and Human-Computer Interaction (GL-IHM).

Djamel Khelladi:
- founding member and co-organiser of the GDR-GPL working group on Software engineering (vélocité logicielle) GT VL.
- founding member and co-organiser of the online seminar series on Engineering of Digital twins EDT.

Olivier Zendra:
- Founding member of the Steering committee of ICOOOLPS workshop
- Member of the Steering committee of HiPEAC CSA

Stéphanie Challita:
- Member of the IEEE Conference Activities Committee

3.6.5 Research Expertise

- Olivier Barais: scientific reviewer for WASP NEST Sweden research program [14]
- Olivier Barais: member of the scientific board of *Pole de compétitivité Image et Réseau* [15]
- Olivier Barais: scientific reviewer for FRQNT- Programme Samuel-de-Champlain (Quebec)
- Olivier Barais: external expert for the H2020 ENACT project
- Olivier Barais: scientific expertise for DGRI international program (20 proposals per year)
- Olivier Barais: AutoActive board of expert (Norvegian project)
- Arnaud Blouin: external ANR (French national research agency) reviewer 2017
- Johann Bourcier : scientific reviewer for CIR (Credit Impot Recherche) 2016-2021.
- Benoit Comebmale: external expert for different national research agencies and the European Union (ERC).
- Olivier Zendra : scientific reviewer for CIR (Credit Impot Recherche) 2020, 2021.
- Olivier Zendra : scientific reviewer for ANR (French national research agency) AAP Générique 2021.
- Olivier Zendra : scientific reviewer for PEC (Pôle d'Excellence Cyber) 2021.

3.6.6 Research Administration

From Jan. 2012 to Dec 2020, Jean-Marc Jézéquel has been Director of IRISA (UMR 6074, one of the largest Computer Science public research lab in France, with over 850 members). He was also Coordinator of the academic club of the French Cyber-defense Excellence Cluster, and Director of the Rennes Node of EIT Digital.

[14] https://wasp-sweden.org/wasp-launches-nest-first-call-open/
[15] https://www.images-et-reseaux.com/

3.7 International Collaborations

Inria Associated Teams and Other Formal Collaborations

- ALE

  Title: *Agile Language Engineering*
  Duration: 2020 - 2023
  Coordinator: Benoit Combemale
  Partners: *Inria (France)*, CWI (Netherlands)
  Inria contact: Benoit Combemale
  Summary: Software engineering faces new challenges with the advent of modern software-intensive systems such as complex critical embedded systems, cyber-physical systems and the Internet of things. Application domains range from robotics, transportation systems, defense to home automation, smart cities, and energy management, among others. Software is more and more pervasive, integrated into large and distributed systems, and dynamically adaptable in response to a complex and open environment. As a major consequence, the engineering of such systems involves multiple stakeholders, each with some form of domain-specific knowledge, and with the increased use of software as an integration layer. Hence more and more organizations are adopting Domain-Specific Languages (DSLs) to allow domain experts to express solutions directly in terms of relevant domain concepts. This new trend raises new challenges about designing DSLs, evolving a set of DSLs and coordinating the use of multiple DSLs for both DSL designers and DSL users. ALE will contribute to the field of Software Language Engineering, aiming to provide more agility to both language designers and language users. The main objective is twofold. First, we aim to help language designers to leverage previous DSL implementation efforts by reusing and combining existing language modules, while automating the deployment of distributed, elastic and collaborative modeling environments. Second, we aim to provide more flexibility to language users by ensuring interoperability between different DSLs, offering live feedback about how the model or program behaves while it is being edited (aka. live programming/modeling), and combining with interactive environments like Jupiter Notebook for literate programming.

- RESIST

  Title: *Resilient Software Science (RESIST)*
  Duration: 2021 - 2024
  Coordinator: Arnaud Gotlieb
  Partners: *Inria (France)*, SIMULA (Norway)
  Inria contact: Mathieu Acher
  Summary: The associated team Resilient Software Science (RESIST) will research the foundations of "*Resilient Software*" *ie.* software systems which can resist failures without significantly degrading their functionality. Over the past years, resilient software systems have become extremely important in various application domains. By combining Inria's and Simula´s combined expertise in software engineering, resilient software, and AI-based software testing, the RESIST associate team will address the following challenges:

    - Utilizing observation data for assessing resilience under varying conditions
    - Digital twins for continuous improvement of resilient autonomous/smart systems
    - Resilient autonomous systems in the digital and physical world

  The goal for RESIST is to explore the science of resilient software by foundational work on advanced a priori testing methods such as metamorphic testing and a posteriori continuous improvements through digital twins.

Informal Collaborations

- Université de Montréal (Canada)
- McGill University (Canada)
- University of Alabama (USA)
- University of Lancaster (UK)
- University of Namur (Belgium)
- Universita degli Studi di Cagliari (Italy)
- Università degli Studi dell'Aquila (Italy)
- JKU Linz (Austria)
- TU Wien (Austria)
- Michigan State University (MSU)
- RWTH Aachen University (Germany)
- KTH (Sweden)

Visits of International Scientists

- Nelly Bencomo, Professor at Durham University, visited the team from November to March 2020.
- Gunter Mussbacher, Professor at McGill University, visited the team from January to June 2020.
- Martin Montperrus, Professor at KTH, Sweden, visited the team in December 2019.
- Nicolas Harrand, PhD Student at KTH, Sweden, visited the team in December 2019.
- Paul Temple, postdoc at University de Namur, visited the team in February 2019.
- Thomas Degueule, postdoc at CWI, visited the team in December 2019
- Alfonso Pierantonio, Associate Professor at Università degli Studi dell'Aquila, visited the team in June 2019
- Mark van den Brand, Professor at Eindhoven University of Technology, visited the team in June 2019
- Yves Le Traon, Professor at the University of Luxembourg, visited the team in June and July 2018, 2019, 2021.
- François Fouquet, Junior Researcher at the SnT (Lux), visited the team in March 2018.
- Jordi Cabot, Research Professor at Internet Interdisciplinary Institute, the Research center of the Open University of Catalonia (UOC), SOM Research Lab leader, visited the team in December 2018.
- Erwan Bousse, postdoctoral researcher at TU Wien, Austria, visited the team from January until Aug 2018
- Marcel Heinz, research assistant and PhD student at University of Koblenz-Landau visited the team in Jul 2018
- Yves Le Traon, Professor at the University of Luxembourg, visited the team in June and July 2017.
- Tanja Mayerhofer, Junior Researcher at the TU Wien, visited the team in March 2017.
- François Fouquet, Junior Researcher at the SnT (Lux), visited the team in November 2017

Visits to International Teams

- Pierre JeanJean visited CWI for 1 week in December 2019 in the context of the Associated Team ALE.
- Benoit Combemale made several short visits at CWI in the context of the Associated Team ALE, visited McGill University in June 2019, and visited TU Eindhoven in November 2019.
- Olivier Barais made several short visits at KTH in the context of a collaboration with Prof Monperrus and Prof Baudry.
- Djamel E. Khelladi made a one week research visit in December 2019 to the DIRO laboratory at University of Montreal, Canada.
- Fabien Coulon visited CWI for 1 week in June 2018 in the context of the Associated Team ALE.

- Manuel Leduc visited CWI for 1 week in December 2018 in the context of the Associated Team ALE.
- Benoit Combemale made several short visits at CWI in the context of the Associated Team ALE, and visited TU Eindhoven in November 2018.
- Olivier Barais and Amine Benelallam made several short visits at KTH in the context of a collaboration with Prof Monperrus and Prof Baudry.
- Manuel Leduc visited CWI for 3 weeks in September 2017
- Benoit Combemale visited Professor Jorg Kienzle at McGill University (Canada) for 3 months in 2017; and made several short visits at CWI (The Netherlands)

4 Funding

National Initiatives

Regional Initiatives

PEC – Pôle d'Excellence Cyber. Coordinator: Université de Rennes 1. Dates: 2016-2019. Formal and Executable Specification of domain-specific language families. ∼ **150 k€**

ANR

MC-Evo$_2$ ANR JCJC. Coordinator: Djamel Khelladi (CNRS). Dates: 2021-2025 Software maintenance represents 40% to 80% of the total cost of developing software. On 65 projects, an IT company reported a cost of several million dollars, with a 25% higher cost on complex projects. Nowadays, software evolves frequently with the philosophy "Release early, release often" embraced by IT giants like the GAFAM. Thus, making software maintenance hard and costly. Developing complex software inevitably requires developers to handle multiple dimensions, such as APIs to use, tests to write, models to reason with, etc. When software evolves, a co-evolution is usually necessary as a follow-up, to resolve the impacts caused by the evolution changes. For example, when APIs evolve, code must be co-evolved, or when code evolves, its tests must be co-evolved. The goals of this project are to: 1) address these challenges from a novel perspective, namely a multidimensional co-evolution approach, 2) investigate empirically the multidimensional co-evolution in practice in GitHub, Maven, and Eclipse, 3) automate and propagate the multidimensional co-evolution between the software code, APIs, tests, and models.

VaryVary ANR JCJC. Coordinator: Mathieu Acher (UR1). Dates: 2017-2021 Most modern software systems are subject to variation or come in many variants. Hundreds of configuration options, features, or plugins can be combined, each potentially with distinct functionality and effects on execution time, memory footprint, etc. Among configurations, some of them are chosen and do not compile, crash at run time, do not pass a test suite, or do not reach a certain performance quality. In this JCJC ANR project, we follow a thought-provocative and unexplored direction: We consider that the variability boundary of a software system can be specialized and should vary when needs be. The goal of this project is to provide theories, methods and techniques to make vary variability.

SOPRANO. Coordinator: CEA. CEA, University of Paris-Sud, Inria Rennes, OcamlPro, Adacore. Dates: 2014-2018. 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.

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

Gdiv MRSE. Coordinator: B. Baudry, Inria Rennes. Dates: 2014-2016 The objective of GDiv is to setup a strong network of European partners around the core team composed of Inria and SINTEF. This network will gather another academic partner and between 3 and 5 industry partners in the areas of software development and deployment. The project proposal setup by the GDiv network will address the risks of large scale software reuse through integrated, multi-level software diversification techniques.

IUF

REVARY. Coordinator: Mathieu Acher (UR1). Dates: 2021-2026. Anyone, engineer or scientist has thousands of options to vary software (e.g. Linux, VLC, Firefox) and adapt it to their needs. The number of possible variants is colossal (generally greater than the number of atoms in the universe). Even worse, the variants are also subject to the variability of the operating system, the compiler, or the data being processed. This deep variability, while fascinating, certainly induces unreasonable complexity: many options have a negligible impact on the software. REVARY aims to find and kill unnecessary variability, allowing to decrease the size and complexity of systems with cascading effects beneficial to execution time, security or energy consumption

DGA

LangComponent (CYBERDEFENSE). Coordinator: DGA. Partners: DGA MI, INRIA. Dates: 2019-2022. 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.

FPML (CYBERDEFENSE).Coordinator: DGA. Partners: DGA MI, Inria. Dates: 2014-2017. 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.

Cominlabs

PROFILE. Coordinator: Université de Rennes 1. Partners: INRIA, Université de Rennes 2. Dates: 2016-2019. The PROFILE project brings together experts from law, computer science and sociology to address the challenges raised by online profiling, following a multidisciplinary approach.

visiometivier@visio.inria.fr

European Projects

H2020 ICT-10-2016 STAMP. Coordinator: Inria Rennes. Other partners: ATOS, ActiveEon, OW2, TellU, Engineering, XWiki, TU Delft, SINTEF. Dates: 2016-2019 Leveraging advanced research in automatic test generation, STAMP aims at pushing automation in DevOps one step further through innovative methods of test amplification. It reuse existing assets (test cases, API descriptions, dependency models), in order to generate more test cases and test configurations each time the application is updated. Acting at all steps of development cycle, STAMP techniques aim at reducing the number and cost of regression bugs at unit level, configuration level and production stage. ∼ **600 k€**

H2020-ICT-2017-1 TeamPlay. Coordinator Inria Rennes. Other partners: Univ. of Bristol (UK), Univ. of St Andrews (UK), Univ. of Amsterdam (NL), Univ. of Southern Denmark (DK), TUHH (DE), AbsInt (DE), Irida Labs (GR), Thales Alenia space - Espana (ES), Skywatch (DK). Dates: (2018-) 2020-2021 in DiverSE. The TeamPlay (Time, Energy and Security Analysis for Multi/Many-core heterogenerous PLAtforms) project aims to develop new, formally-motivated, techniques that will allow execution time, energy usage, security, and other important non-functional properties of parallel software to be treated effectively, and as first- class citizens. We build this into a toolbox for developing highly parallel software for low-energy systems, as required by the internet of things, cyber-physical systems etc. The TeamPlay approach allows programs to reflect directly on their own time, energy consumption, security, etc., as well as enabling the developer to reason about both the functional and the non-functional properties of their software at the source code level. ∼ **620 k€**

H2020-ICT-2019-2 HiPEAC 6. Coordinator Univ. Ghent. Partners: Barcelona Supercomputing Center (ES), Chalmers Univ. (SE), FORTH (GR), Inria Rennes (FR), Univ. of Edinburgh (UK), Univ. of Pisa (IT), RWTH Aachen (DE), CEA (FR), ARM (UK), Thales (FR), ARTEMIS-IA (NL). Dates (2019-) 2020-2023 in DiverSE. HiPEAC (High Performance Embedded Architecture and Compilation) is the premier focal point for networking, dissemination, training, and collaboration activities in Europe for researchers, industry, and policy related to computing systems. Today, its network, the biggest of its kind in Europe, numbers over 2,000 specialists. HiPEAC's mission is to advance computer architecture and computing systems research and development as a discipline in Europe. ∼ **35 k€**

EIT Digital UAV-Retina. Unmanned Aerial Vehicles - Retina. Duration: from 2018-10 to 2019-12. Coordinator: UR1. Other partners: Bright Cape (company from the Netherlands), FBK (Trento University, Italy), JCP Connect (company from France), Tellus Environment (company from France), Fire Department of Ille et Vilaine (France), Fire Department of Trento (Italy). The UAV-Retina objectives aims at the creation of s startup company for a flexible autonomous drone platform for search and rescue, using advanced unmanned vehicles and data analytics. The markets for the company will be Improved Explosive Devices detection, support for firefighting operations, and support for avalanche search and rescue operations. ∼ **150 k€**

CEF AQMO. Coordinator: UR1. Airbreizh - Surveillance of Brittany air quality association, branch of the French national air surveillance network, AmpliSIM - SME specialized in pollution numerical simulation, CNRS / IDRIS, GENCI, Neovia, Rennes Métropole, Ryax Technologies. Dates: 2018-2020. The AQMO project is co-financed by the European Union through its CEF programme. It addresses the air quality challenge, thanks to the development of a smart city pilot in the area of Rennes Metropole. The project will provide an end-to-end urban platform that extends current practices in air quality measurements. The AQMO platform will provide citizens, local authorities, scientific organizations and private companies with new data and innovative services based on computing simulation. ∼ **150 k€**

FP7 STREP HEADS. Coordinator: SINTEF. Other partners: Inria, Software AG, ATC, Tellu, eZmonitoring. Dates: 2013-2017. 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 to their users. ∼ **600 k€**

FP7 FET STREP DIVERSIFY. Coordinator: Inria (DIVERSE). Partners: SINTEF, Université de Rennes 1, Trinity College Dublin, Inria (DiverSE, SPIRALS). Dates: 2013-2016. 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. ∼ **500 k€**

ICT COST Action MPM4CPS (IC1404). Chair of the Action: Prof Hans Vangheluwe (BE). Dates: 2014-2018. 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. 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). 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. $\sim$ **50 k€**

Industrial Contracts

Competitivity Clusters/ BGLE / LEOC

Occiware. Coordinator: Open Wide. Partners: 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: 2014-2018. The Occiware project aims to establish a formal and equipped framework for the management of all cloud resource based on the OCCI standard. $\sim$ **300 k€**

CLARITY. Coordinator: Obeo. 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. 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). $\sim$ **500 k€**

BGLE2 CONNEXION. Coordinator: EDF. Other partners: Atos WorldGrid, Rolls-Royce Civil Nuclear, Corys TESS, Esterel Technologies, All4Tec, Predict, CEA, Inria, CNRS / CRAN, ENS Cachan, LIG, Telecom ParisTech. Dates: 2012-2016 The cluster CONNEXION aims to propose and validate an innovative architecture platforms suitable control systems for nuclear power plants in France and abroad. In this project the 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. $\sim$ **300 k€**

Bilateral Contracts with Industry

ADR Nokia. Coordinator: Inria. Dates: 2017-2021. The goal of this project is to integrate a chaos engineering principles to IoT Services frameworks to improve the robustness of the software-defined network services using this approach and to explore the concept of equivalence for software-defined network services and propose an approach to constantly evolve the attack surface of the network services. $\sim$ **180 k€**

BCOM. Coordinator: UR1. Dates: 2018-2024. The aim of the Falcon project is to investigate how to improve the resale of available resources in private clouds to third parties. In this context, the collaboration with DiverSE mainly aims at working on efficient techniques for the design of consumption models and resource consumption forecasting models. These models are then used as a knowledge base in a classical autonomous loop. $\sim$ **10 k€**

GLOSE. Partners: Inria/CNRS/Safran. Dates: 2017-2021. The GLOSE project develops new techniques for heterogeneous modeling and simulation in the context of systems engineering. It aims to provide formal and operational tools and methods to formalize the behavioral semantics of the various modeling languages used at system-level. These semantics will be used to extract behavioral language interfaces supporting the definition of coordination patterns. These patterns, in turn, can systematically be used to drive the coordination of any model conforming to these languages. The project is structured according to the following tasks: concurrent xDSML engineering, coordination of discrete models, and coordination of discrete/continuous models. The project is funded in the context of the network DESIR, and supported by the GEMOC initiative. $\sim$ **350 k€**

GLOSE Demonstrator. Partners: Inria/Safran. Dates: 2019-2020. Demonstrator illustrating the technologies involved in the WP5 off the GLOSE project. The use case chosen for the demonstrator is the high-level description of a remote control drone system, whose the main objective is to illustrate the design and simulation of the main functional chains, the possible interactivity with the model in order to raise the level of understanding over the models built, and possibly the exploration of the design space. $\sim$ **100 k€**

OneShotSoftware. Partners: Inria/Orange. Dates: 2017-2019. The OSS project investigates an extreme version of moving target defense where a slightly different version of the application is deployed each time it is used (e.g., for crypto functions or payment services). We investigate the analysis, synthesis and transformation techniques to support diversification at 5 points of a software construction pipeline, which, once combined yield up to billions of variants. We also evaluate the support of diversification as a first class property in DevOps. $\sim$ **250 k€**

Debug4Science. Partners: Inria/CEA DAM. Dates: 2020-2022. Debug4Science aims to propose a disciplined approach to develop domain-specific debugging facilities for Domain-Specific Languages within the context of scientific computing and numerical analysis. Debug4Science is a bilateral collaboration (2020-2022), between the CEA DAM/DIF and the DiverSE team at Inria. $\sim$ **250 k€**

Agileo. Partners: Inria/Agileo. Dates: 2017-2018 In this project we mainly design a systematic mapping study on modeling for Industry 4.0 in order to share a common scientific roadmap. $\sim$ **10 k€**

Kereval. Partners: INSA Rennes/Kereval. Dates: 2019-2020. Front-ends testing in a DevOps context, Romain Lebouc's PhD Cifre project. $\sim$ **20 k€**

Obeo. Partners: Inria/Obéo. Dates: 2017-2020. Web engineering for domain-specific modeling languages, Fabien Coulon's PhD Cifre project. $\sim$ **150 k€**

OKWind. Partners: UR1/OKWind. Dates: 2017-2020. Models@runtime to improve self-consumption of renewable energies, Alexandre Rio's PhD Cifre project. $\sim$ **150 k€**

Orange. Partners: UR1/Orange. Dates: 2016-2019. Modelling and evaluating security of authentication paths, Youssou Ndiaye's PhD Cifre project. $\sim$ **150 k€**

Orange. Partners: UR1/Orange. Dates: 2020-2023. Context aware adaptive authentification, Anne Bumiller's PhD Cifre project. $\sim$ **150 k€**

Keolis. Partners: UR1/Keolis. Dates: 2018-2021. Urban mobility: machine learning for building simulators using large amounts of data, Gauthier LYAN's PhD Cifre project. $\sim$ **150 k€**

FaberNovel. Partners: UR1/FaberNovel. Dates: 2018-2021. Abstractions for linked data and the programmable web, Antoine Cheron's PhD Cifre project. $\sim$ **150 k€**

Inria Project Labs, Exploratory Research Actions and Technological Development Actions

No support provided by INRIA

Associated teams and other international projects

Inria International Labs

Simula-Inria RESIST. Associated team. http://gemoc.org/resist/ Title: Resilient Software Science. International Partner: Simula (Normay). Start year: 2021. The goal of the RESIST associate team is to explore the science of resilient software by foundational work on advanced a priori testing methods such as metamorphic testing and a posteriori continuous improvements through digital twins. $\sim$ **20 k€**

IIL CWI-Inria ALE. Associated team. http://gemoc.org/ale Title: Agile Language Engineering. International Partner: CWI (Netherlands) with Tijs van der Storm. Start year: 2017. ALE contributes to the field of Software Language Engineering, aiming to provide more agility to both language designers and language users. The main objective is twofold. First, we aim to help language designers to leverage previous DSL implementation efforts by reusing and combining existing language modules. Second, we aim to provide more flexibility to language users by ensuring interoperability between different DSLs and offering live feedback about how the model or program behaves while it is being edited (aka. live programming/modeling). ∼ **30 k€**

Collaborations with major European organizations

Vipo Project. Vipo is an innovation project from EIT Digital. This year, we bring our expertise on native cloud architecture and adaptable architectures for this project. ∼ **60 k€**

Participation in Other International Programs

The **GEMOC studio** has been sustained through the creation of a Research Consortium at the Eclipse Foundation.

International initiative GEMOC. The GEMOC initiative (cf. http://www.gemoc.org) is an open and international initiative launched in 2013 that coordinates research partners worldwide to develop breakthrough software language engineering (SLE) approaches for 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. 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.

5 Objectives for the Next Four Years

5.1 Context

Applications are becoming more complex and the demand for faster development is increasing. In order to better adapt to the unbridled evolution of requirements in markets where software plays an essential role, companies are changing the way they design, develop, secure and deploy applications, by relying on:

- A massive use of reusable libraries from a rich but fragmented eco-system;
- An increasing configurability of most of the produced software;
- A strongly increasing evolution frequency;
- Cloud-native architectures based on containers, naturally leading to a diversity of programming languages used, and to the emergence of infrastructure, dependency, project and deployment descriptors (models);
- Implementations of fully automated software supply chains;
- The use of lowcode/nocode platforms;
- The use of ever richer IDEs, more and more deployed in SaaS mode;
- The massive use of data and artificial intelligence techniques in software production chains.

These trends are set to continue, all the while ensuring the security properties of the produced and distributed software.

The numbers in the examples below help understand why this evolution of modern software engineering brings a **change of dimension**:

- When designing a simple kitchen sink (*hello world*) with the `angular` framework, more than 1600 dependencies of JavaScript libraries are pulled.
- The numbers revealed by google in 2018 showed that over 500 million tests are run *per day* inside Google's systems, leading to over 4 millions daily builds.
- Also at Google, they reported 86TB of data, including two billion lines of code in nine million source files [64]. Their software also rapidly evolves both in term of frequency and in term of size. Again, at Google, 25,000 developers typically commit 16,000 changes to the codebase on a single workday. This is also the case for most of software code, including open source software.
- x264, a highly popular and configurable video encoder, provides 100+ options that can take boolean, integer or string values. There are different ways of compiling x264, and it is well-known that the compiler options (e.g., -O1 –O2 –O3 of gcc) can influence the performance of a software; the widely used gcc compiler, for example, offers more than 200 options. The x264 encoder can be executed on different configurations of the Linux operating system, whose options may in turn influence x264 execution time; in recent versions (> 5), there are 16000+ options to the Linux kernel. Last but not least, x264 should be able to encode lots of different videos, in different formats and with different visual properties, therefore the variability of the input space is also huge. Overall, the variability space is enormous, and ideally x264 should be run and tested in all these settings. But a rough estimation shows that the number of possible configurations, resulting from the combination of the different variability layers, is 10^{6000}.

The DiverSE research project is working and evolving in the context of this acceleration. We are active at all stages of the **software supply chain**. Software supply chain covers all the activities and stakeholders that relate to software production and delivery. All these activities and stakeholders have to be smartly managed together as part of an overall strategy. The goal of supply chain management (SCM) is to meet customer demands with the most efficient use of resources possible.

In this context, DiverSE is particularly interested in the following research questions:

- How to engineer tool-based abstractions for a given set of experts in order to foster their socio-technical collaboration;

- How to generate and exploit useful data for the optimization of this supply chain, in particular for the control of variability and the management of the co-evolution of the various software artifacts;
- How to increase the confidence in the produced software, by working on the resilience and security of the artifacts produced throughout this supply chain.

5.2 Research Directions

DiverSE will continue to explore *Software Diversity* over the next period. Leveraging our strong background on Model-Driven Engineering, and our large expertise on several related fields (programming languages, distributed systems, GUI, machine learning, security...), *we explore tools and methods to embrace the inherent diversity in software engineering*, from the stakeholders and underlying tool-supported languages involved in the software system life cycle, to the configuration and evolution space of the modern software systems, and the heterogeneity of the targeted execution platforms. Hence, we organize our research directions over the next period according to three axes (cf. Fig. 3):

- **Axis #1: Software Language Engineering.** We explore the future engineering and scientific environments to support the socio-technical coordination among the various stakeholders involved across modern software system life cycles.
- **Axis #2: Spatio-temporal Variability in Software and Systems.** We explore systematic and automatic approaches to cope with software variability, both in space (software variants) and time (software maintenance and evolution).
- **Axis #3: DevSecOps and Resilience Engineering for Software and Systems.** We explore smart continuous integration and deployment pipelines to ensure the delivery of secure and resilient software systems on heterogeneous execution platforms (cloud, IoT. . .).

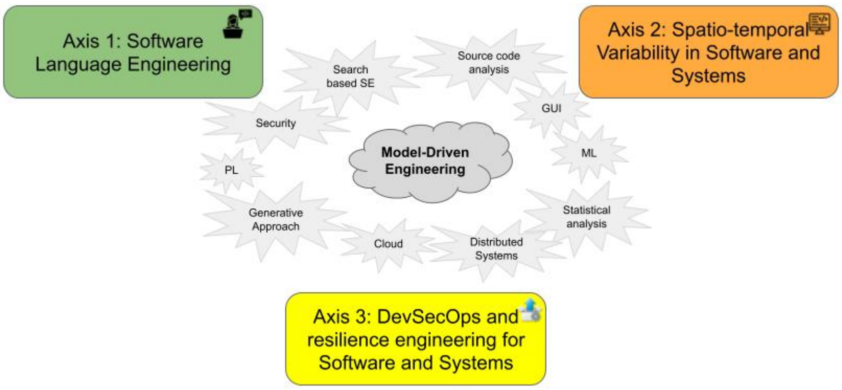

Figure 3: The three future research axes of DiverSE, relying on model driven engineering scientific background and leveraging several related fields

5.2.1 Axis #1: Software Language Engineering

Overall objective. The disruptive design of new, complex systems requires a high degree of flexibility in the communication between many stakeholders, often limited by the silo-like structure of the organization itself (cf. Conway's law). To overcome this constraint, modern engineering environments aim to: (i) better manage the necessary exchanges between the different stakeholders; (ii) provide a unique and usable place for information sharing; and (iii) ensure the consistency of the many points of view. Software languages are the key pivot between the *diverse* stakeholders involved, and the software systems they have to implement. Domain-Specific (Modeling) Languages enable stakeholders to address the *diverse* concerns through specific points of view, and their coordinated use is essential to support the socio-technical coordination across the overall software system life cycle. Our perspectives on Software Language Engineering over the next period is presented in Figure 4 and detailed in the following paragraphs.

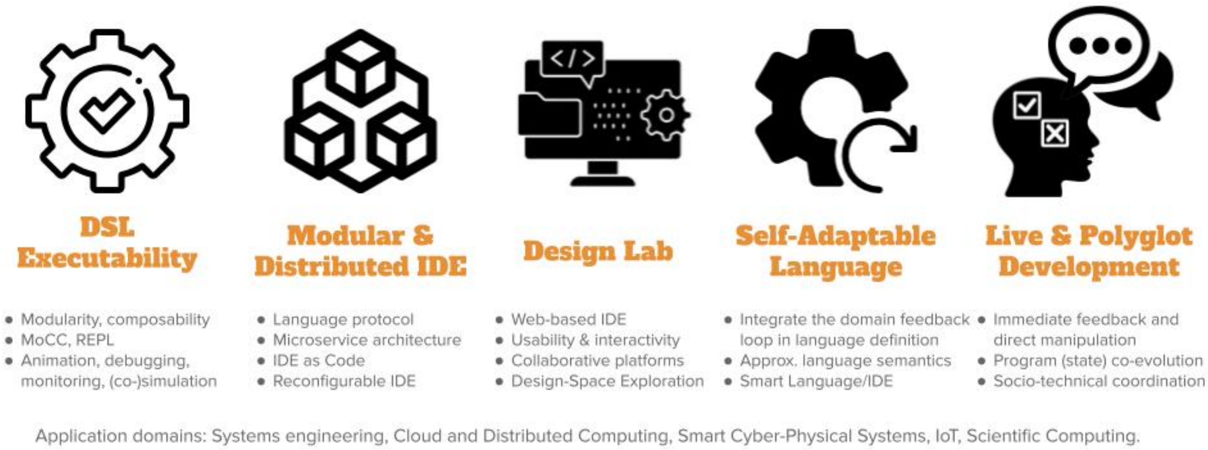

Figure 4: Perspectives on Software Language Engineering (axis #1)

DSL Executability. Providing rich and adequate environments is key to the adoption of domain-specific languages. In particular, we focus on tools that support model and program execution. We explore the foundations to define the required concerns in language specification, and systematic approaches to derive environments (*e.g.,* IDE, notebook, design labs) including debuggers, animators, simulators, loggers, monitors, trade-off analysis, etc.

Modular & Distributed IDE. IDEs are indispensable companions to software languages. They are increasingly turning towards Web-based platforms, heavily relying on cloud infrastructures and forges. Since all language services require different computing capacities and response times (to guarantee a user-friendly experience within the IDE) and use shared resources (*e.g.,* the program), we explore new architectures for their modularization and systematic approaches for their individual deployment and dynamic adaptation within an IDE. To cope with the ever-growing number of programming languages, manufacturers of Integrated Development Environments (IDE) have recently defined protocols as a way to use and share multiple language services in language-agnostic environments. These protocols rely on a proper specification of the services that are commonly found in the tool support of general-purpose languages, and define a fixed set of capabilities to offer in the IDE. However, new languages regularly appear offering unique constructs (e.g., DSLs), and supported by dedicated services to be offered as new capabilities in IDEs. This trend leads to the multiplication of new protocols, hard to combine and possibly incompatible (e.g., overlap, different technological stacks). Beyond the proposition of specific protocols, we will explore an original approach to be able to specify language protocols and to offer IDEs to be configured with such protocol specifications. IDEs went from directly supporting languages to protocols, and we envision the next step: *IDE as code*, where language protocols are created or inferred on demand and serve as support of an adaptation loop taking in charge of the (re)configuration of the IDE.

Design Lab. Web-based and cloud-native IDEs open new opportunities to bridge the gap between the IDE and collaborative platforms, *e.g.,* forges. In the complex world of software systems, we explore new approaches to reduce the distance between the various stakeholders (e.g., systems engineers and all those involved in specialty engineering) and to improve the interactions between them through an adapted tool chain. We aim to improve the usability of development cycles with efficiency, affordance and satisfaction. We also explore new approaches to explore and interact with the design space or other concerns such as software and systems human values or security, and provide facilities for trade-off analysis and decision making.

Live & Polyglot Development. As of today, polyglot development is massively popular and virtually all software systems put multiple languages to use, which not only complexifies their development, but also their evolution and maintenance. Moreover, as software grows more critical in new application domains (e.g., data analytics, health or scientific computing), it is crucial to ease the participation of scientists, decision-makers, and more generally non-software experts. Live programming makes it possible to change a program while it is running, by propagating changes on a program code to its run-time state. This effectively bridges the gulf of evaluation between program writing and program execution: the effects a change has on the running system are immediately visible, and the developer can take immediate action. The challenges at the intersection of polyglot and live programming have received little attention so far, and we envision a language design and implementation approach to specify domain-specific languages and their coordination, and automatically provide interactive domain-specific environments for live and polyglot programming.

Self-Adaptable Language. Over recent years, self-adaptation has become a concern for many software systems that operate in complex and changing environments. At the core of self-adaptation lies a feedback loop and its associated trade-off reasoning, to decide on the best course of action. However, existing software languages do not abstract the development and execution of such feedback loops for self-adaptable systems. Developers have to fall back to ad-hoc solutions to implement self-adaptable systems, often with wide-ranging design implications (e.g., explicit MAPE-K loop). Furthermore, existing software languages do not capitalize on monitored usage data of a language and its modeling environment. This hinders the continuous and automatic evolution of a software language based on feedback loops from the modeling environment and runtime software system. To address the aforementioned issues, we will explore the concept of Self-Adaptable Language (SAL) to abstract the feedback loops at both system and language levels.

5.2.2 Axis #2: Spatio-temporal Variability in Software and Systems

Overall objective. Leveraging on our longstanding activity on variability management for software product lines and configurable systems covering *diverse* scenarios of use, we will investigate over the next period the impact of such a variability across the *diverse* layers, incl. source code, input/output data, compilation chain, operating systems and underlying execution platforms. We envision a better support and assistance for the configuration and optimisation (e.g., non-functional properties) of software systems according to this deep variability. Moreover, as software systems involve *diverse* artefacts (*e.g.,* APIs, tests, models, scripts, data, cloud services, documentation, deployment descriptors...), we will investigate their continuous co-evolution during the overall lifecycle, including maintenance and evolution. Our perspectives on spatio-temporal variability over the next period is presented in Figure 5 and is detailed in the following paragraphs.

Deep Software Variability. Software systems can be configured to reach specific functional goals and non-functional performance, either statically at compile time or through the choice of command line options at runtime. We observed that considering the software layer only might be a naive approach to tune the performance of the system or to test its functional correctness. In fact, many layers (hardware, operating system, input data, etc.), which are themselves subject to variability, can alter the performances or functionalities of software configurations. We call *deep software variability* the interaction of all variability layers that could modify the behavior or

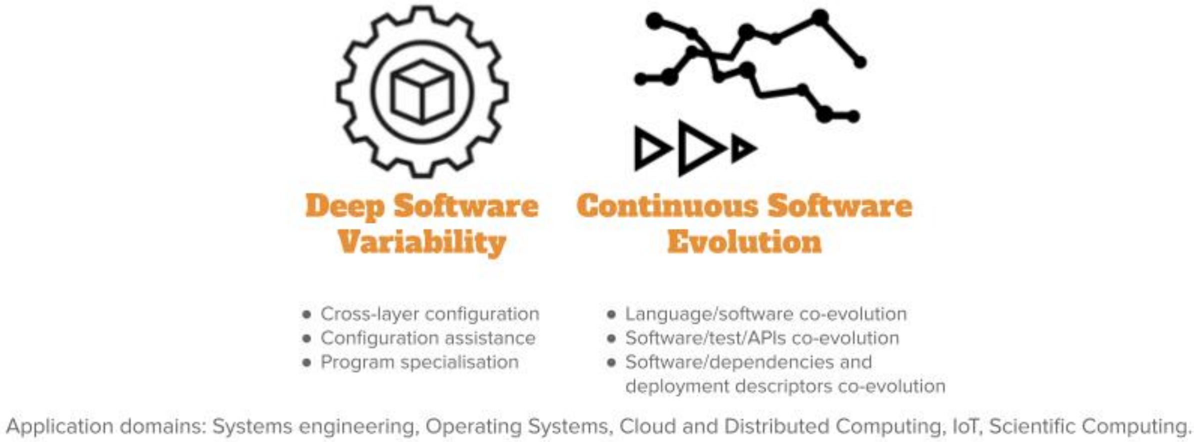

Figure 5: Perspectives on Spatio-temporal Variability in Software and Systems (axis #2)

non-functional properties of a software. Deep software variability calls to investigate how to systematically handle cross-layer configuration. The diversification of the different layers is also an opportunity to test the robustness and resilience of the software layer in multiple environments. Another interesting challenge is to tune the software for one specific executing environment. In essence, deep software variability questions the generalization of the configuration knowledge.

Continuous Software Evolution. Nowadays, software development has become more and more complex, involving various artefacts, such as APIs, tests, models, scripts, data, cloud services, documentation, etc., and embedding millions of lines of code (LOC). Recent evidence highlights continuous software evolution based on thousands of commits, hundreds of releases, all done by thousands of developers. We focus on the following essential backbone dimensions in software engineering: languages, models, APIs, tests and deployment descriptors, all revolving around software code implementation. We will explore the foundations of a multidimensional and polyglot co-evolution platform, and will provide a better understanding with new empirical evidence and knowledge.

5.2.3 Axis #3: DevSecOps and Resilience Engineering for Software and Systems

Overall objective. The production and delivery of modern software systems involves the integration of *diverse* dependencies and continuous deployment on *diverse* execution platforms in the form of large distributed socio-technical systems. This leads to new software architectures and programming models, as well as complex supply chains for final delivery to system users. In order to boost cybersecurity, we want to provide strong support to software engineers and IT teams in the development and delivery of secure and resilient software systems, ie. systems able to resist or recover from cyberattacks. Our perspectives on DevSecOps and Resilience Engineering over the next period are presented in Figure 6 and detailed in the following paragraphs.

Secure & Resilient Architecture. Continuous integration and deployment pipelines are processes implementing complex software supply chains. We envision an explicit and early consideration of security properties in such pipelines to help detecting vulnerabilities. In particular, we integrate the security concern in Model-Based System Analysis (MBSA) approaches, and explore guidelines, tools and methods to drive the definition of secure and resilient architectures. We also investigate resilience at runtime through frameworks for autonomic computing and data-centric applications, both for the software systems and the associated deployment descriptors.

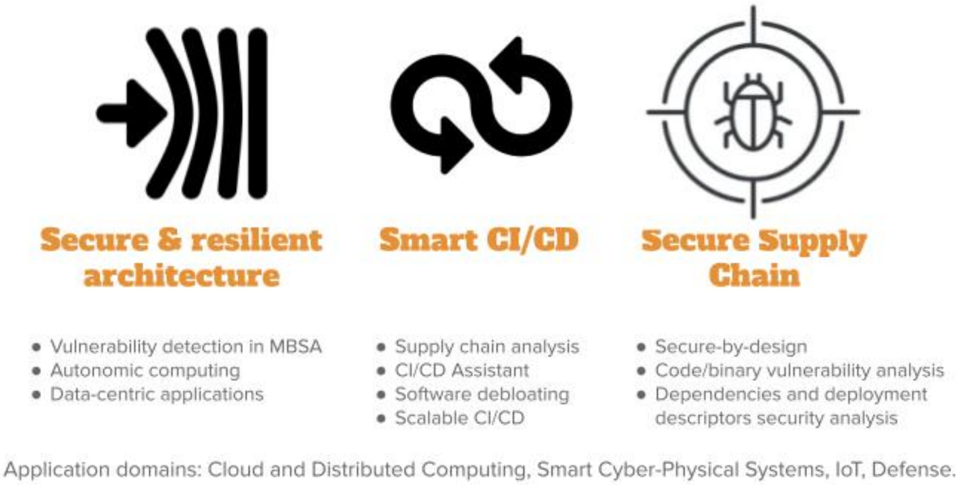

Figure 6: Perspectives on DevSecOps and Resilience Eng. for Software and Systems (axis #3)

Smart CI/CD. Dependencies management, Infrastructure as Code (IaC) and DevOps practices open opportunities to analyze complex supply chains. We aim at providing relevant metrics to evaluate and ensure the security of such supply chains, advanced assistants to help in specifying corresponding pipelines, and new approaches to optimize them (*e.g.,* software debloating, scalability...). We study how supply chains can actively leverage software variability and diversity to increase cybersecurity and resilience.

Secure Supply Chain. In order to produce secure and resilient software systems, we explore new secure-by-design foundations that integrate security concerns as first class entities through a seamless continuum from the design to the continuous integration and deployment. We explore new models, architectures, inter-relations, and static and dynamic analyses that rely on explicitly expressed security concerns to ensure a secure and resilient supply chain. We lead research on automatic vulnerability and malware detection in modern supply chains, considering the various artefacts either as white boxes enabling source code analysis (to avoid accidental vulnerabilities or intentional ones or code poisoning), or as black boxes requiring binary analysis (to find malware or vulnerabilities). We also conduct research activities in dependencies and deployment descriptors security analysis.

DiverSE's ID:

The **targeted stakeholder** of our research are :

- Software developers and Ops
- Software or System Architects
- Domain experts

Our **application domains** are:

- SE for Cloud native/WE/IoT: Future Internet Continuum (Axis 1, 2 and 3)
  - BCOM, Nokia, CISCO
- SE for Systems engineering (avionics, cyber,...) (Axis 1 and 3)
  - Obéo, Airbus, Safran, DGA, Keolis, OKWind
- SE for Scientific Software (Axis 1)
  - CEA, Safran, OSUR

Our **objects of research** are:

- Large scale, highly configurable open source software (Axis 1, 2, 3)
  - Linux kernel, JHipster, TAC

- Modern IDE and Virtual Lab (Axis 1, 3)
  - Vscode, eclipse/eclipse theia, jupyter
- Industrial Engineering platform (Axis 1, 3)
  - Capella (Thales), CosApp (Safran), NabLab (CEA)

5.3 Project implementation

This section discusses three major points:

- Why is it relevant that all these scientific challenges are jointly conducted within the same team?
- Why have we gathered the best team of researchers to address these scientific challenges?
- Why do we have the human and financial resources to address these challenges?

Why is it relevant that all these scientific challenges are jointly conducted within the same team?

Most of the scientific challenges crosscut several axes, and many the different industrial projects that we carry out benefit from the results from one of the other axes to carry out their work. Each permanent member works on at least two of the sub-axes presented in the section above.

Why have we gathered the best team of researchers to address these scientific challenges?

The DiverSE team offers several interesting features:

- We bring together a rather unique combination of skills, associating a very high technical expertise, a scientific excellence recognized worldwide in the field of software engineering, a desire to lead the scientific community, and a credibility with our industrial partners.
- The DiverSE team members share a common scientific background in the field of software modeling, and more generally a strong scientific and technical culture in software engineering.
- We share the same values: scientific excellence, technical expertise, reproducible science, community leadership, strong interaction with industry producing a high level of synergies (mixing scientific curiosity and a strong interest in industrial problems), PhD students training, team spirit.
- We are working in a friendly atmosphere, enjoying sharing coffees.

Why do we have the human and financial resources to address these challenges?

All research activities are already partially funded and integrated in international initiatives. DiverSE is large enough to deal with the departure or arrival of members, even at mid-project: the turbulent experience in terms of human resources between 2017 and 2020 has proven the resilience of the group.

Additional resources

Reviewers could find in the appendix of this document (following the bibliography), a SWOT analysis of the team (see appendix A) and a short report on the team's scientific background (see appendix B).

6 Bibliography of the Project-Team

Doctoral Dissertations and Habilitation Theses

[1] F. J. Acosta Padilla, Self-adaptation for Internet of things applications, Theses, Université Rennes 1, December 2016, [hal:tel-01426219].

[2] G. Bécan, Metamodels and feature models : complementary approaches to formalize product comparison matrices, Theses, Université Rennes 1, September 2016, [hal:tel-01416129].

[3] S. Ben Nasr, Mining and Modeling Variability from Natural Language Documents: Two Case Studies, Theses, Université Rennes 1, April 2016, [hal:tel-01388392].

[4] A. Blouin, Contribution to the Engineering of User Interfaces, Habilitation à diriger des recherches, Université de Rennes 1 [UR1], August 2019, [hal:tel-02354530].

[5] J. Bourgeois, Interactive Demand-Shifting in the Context of Domestic Micro-Generation, Theses, The Open University ; Université de Rennes 1 [UR1], June 2016, [hal:tel-01385022].

[6] M. Boussaa, Automatic non-functional testing and tuning of configurable generators, Theses, Université Rennes 1, September 2017, [hal:tel-01598821].

[7] K. Corre, User controlled trust and security level of Web real-time communications, Theses, Université Rennes 1, May 2018, [hal:tel-01943728].

[8] J.-E. Dartois, Leveraging Cloud unused heterogeneous resources for applications with SLA guarantees, Theses, Univ-Rennes1, September 2020, [hal:tel-03009816].

[9] T. Degueule, Composition and Interoperability for External Domain-Specific Language Engineering, Theses, Université de Rennes 1 [UR1], December 2016, [hal:tel-01427009].

[10] P. Laperdrix, Browser Fingerprinting : Exploring Device Diversity to Augment Authentification and Build Client-Side Countermeasures, Theses, INSA de Rennes, October 2017, [hal:tel-01729126].

[11] G. Le Moulec, Model driven synthesis of virtual reality applications, Theses, INSA de Rennes, September 2018, [hal:tel-01959918].

[12] M. Leduc, On modularity and performance of External Domain-Specific Language implementations, Theses, Université de rennes 1, December 2019, [hal:tel-02418676].

[13] D. F. Méndez Acuña, Leveraging software product lines engineering in the construction of domain specific languages, Theses, Université Rennes 1, December 2016, [hal:tel-01427187].

[14] L. Mouline, Towards a modelling framework with temporal and uncertain data for adaptive systems, Theses, Université Rennes 1 ; Université du Luxembourg, November 2019, [hal:tel-02527656].

[15] Y. Ndiaye, Modeling and assessing the security of the authentication procedure, Theses, Université Rennes 1, December 2019, [hal:tel-02921435].

[16] P. Temple, Investigate the matrix : leveraging variability to specialize software and test suites, Theses, Université Rennes 1, December 2018, [hal:tel-02125962].

[17] O. L. Vera-Pérez, Dynamic program analysis for suggesting test improvements to developers, Theses, Université de Rennes 1 [UR1], December 2019, [hal:tel-02459572].

Articles in International Journals

[18] M. Acher, R. E. Lopez-Herrejon, R. Rabiser, Teaching Software Product Lines: A Snapshot of Current Practices and Challenges, *ACM Transactions of Computing Education*, May 2017, [hal:hal-01522779].

[19] M. Alférez, M. Acher, J. A. Galindo, B. Baudry, D. Benavides, Modeling Variability in the Video Domain: Language and Experience Report, *Software Quality Journal 27*, 1, January 2019, p. 307–347, [doi:10.1007/s11219-017-9400-8], [hal:hal-01688247].

[20] M. Bagherzadeh, K. Jahed, B. Combemale, J. Dingel, Live Modeling in the Context of State Machine Models and Code Generation, *Software and Systems Modeling*, 2020, p. 1–44, [doi:10.1007/s10270-020-00829-y], [hal:hal-02942374].

[21] S. Ben Nasr, G. Bécan, M. Acher, J. F. F. Bosco, N. Sannier, B. Baudry, J.-M. Davril, Automated Extraction of Product Comparison Matrices From Informal Product Descriptions, *Journal of Systems and Software 124*, 2017, p. 82 – 103, [doi:10.1016/j.jss.2016.11.018], [hal:hal-01427218].

[22] A. Benelallam, A. Gómez, M. Tisi, J. Cabot, Distributing Relational Model Transformation on MapReduce, *Journal of Systems and Software 142*, April 2018, p. 1–20, [doi:10.1016/j.jss.2018.04.014], [hal:hal-01863885].

[23] A. Blouin, J.-M. Jézéquel, Interacto: A Modern User Interaction Processing Model, *IEEE Transactions on Software Engineering*, 2021, p. 1–20, [doi:10.1109/TSE.2021.3083321], [hal:hal-03231669].

[24] A. Blouin, V. Lelli, B. Baudry, F. Coulon, User Interface Design Smell: Automatic Detection and Refactoring of Blob Listeners, *Information and Software Technology 102*, May 2018, p. 49–64, [doi:10.1016/j.infsof.2018.05.005], [hal:hal-01499106].

[25] M. Boussaa, O. Barais, G. Sunyé, B. Baudry, Leveraging metamorphic testing to automatically detect inconsistencies in code generator families, *Journal of : Software Testing, Verification and Reliability*, December 2019, [doi:10.1002/stvr.1721], [hal:hal-02422437].

[26] E. Bousse, D. Leroy, B. Combemale, M. Wimmer, B. Baudry, Omniscient Debugging for Executable DSLs, *Journal of Systems and Software 137*, March 2018, p. 261–288, [doi:10.1016/j.jss.2017.11.025], [hal:hal-01662336].

[27] E. Bousse, T. Mayerhofer, B. Combemale, B. Baudry, Advanced and efficient execution trace management for executable domain-specific modeling languages, *Software and Systems Modeling*, February 2019, p. 1–37, [doi:10.1007/s10270-017-0598-5], [hal:hal-01614377].

[28] J.-M. Bruel, B. Combemale, E. M. Guerra, J.-M. Jézéquel, J. Kienzle, J. De Lara, G. Mussbacher, E. Syriani, H. Vangheluwe, Comparing and Classifying Model Transformation Reuse Approaches across Metamodels, *Software and Systems Modeling 19*, 2, 2020, p. 441–465, [doi:10.1007/s10270-019-00762-9], [hal:hal-02317864].

[29] S. Challita, F. Korte, J. Erbel, F. Zalila, J. Grabowski, P. Merle, Model-Based Cloud Resource Management with TOSCA and OCCI, *Software and Systems Modeling*, February 2021, p. 1–23, [doi:10.1007/s10270-021-00869-y], [hal:hal-03122452].

[30] B. Combemale, J. Kienzle, G. Mussbacher, H. Ali, D. Amyot, M. Bagherzadeh, E. Batot, N. Bencomo, B. Benni, J.-M. Bruel, J. Cabot, B. H. C. Cheng, P. Collet, G. Engels, R. Heinrich, J.-M. Jézéquel, A. Koziolek, S. Mosser, R. Reussner, H. Sahraoui, R. Saini, J. Sallou, S. Stinckwich, E. Syriani, M. Wimmer, A Hitchhiker's Guide to Model-Driven Engineering for Data-Centric Systems, *IEEE Software*, 2020, p. 9, [doi:10.1109/MS.2020.2995125], [hal:hal-02612087].

[31] B. Combemale, J. Kienzle, G. Mussbacher, O. Barais, E. Bousse, W. Cazzola, P. Collet, T. Degueule, R. Heinrich, J.-M. Jézéquel, M. Leduc, T. Mayerhofer, S. Mosser, M. Schöttle, M. Strittmatter, A. Wortmann, Concern-Oriented Language Development (COLD): Fostering Reuse in Language Engineering, *Computer Languages, Systems and Structures 54*, 2018, p. 139–155, [doi:10.1016/j.cl.2018.05.004], [hal:hal-01803008].

[32] K. Corre, O. Barais, G. Sunyé, V. Frey, J.-M. Crom, Why can't users choose their identity providers on the web?, *Proceedings on Privacy Enhancing Technologies 2017*, 3, January 2017, p. 72–86, [doi:10.1515/popets-2017-0029], [hal:hal-01611048].

[33] K. Corre, S. Bécot, O. Barais, G. Sunyé, A WebRTC Extension to Allow Identity Negotiation at Runtime, *Lecture Notes in Computer Science 10360*, 2017, p. 412–419, [doi:10.1007/978-3-319-60131-1_27], [hal:hal-01611057].

[34] B. Danglot, M. Monperrus, W. Rudametkin, B. Baudry, An approach and benchmark to detect behavioral changes of commits in continuous integration, *Empirical Software Engineering 25*, 4, July 2020, p. 2379–2415, [doi:10.1007/s10664-019-09794-7], [hal:hal-03121735].

[35] B. Danglot, P. Preux, B. Baudry, M. Monperrus, Correctness Attraction: A Study of Stability of Software Behavior Under Runtime Perturbation, *Empirical Software Engineering 23*, 4, August 2018, p. 2086–2119, https://arxiv.org/abs/1611.09187, [doi:10.1007/s10664-017-9571-8], [hal:hal-01378523].

[36] B. Danglot, O. L. Vera-Pérez, B. Baudry, M. Monperrus, Automatic Test Improvement with DSpot: a Study with Ten Mature Open-Source Projects, *Empirical Software Engineering*, 2019, p. 1–35, [doi:10.1007/s10664-019-09692-y], [hal:hal-01923575].

[37] B. Danglot, O. L. Vera-Pérez, Z. Yu, A. Zaidman, M. Monperrus, B. Baudry, A Snowballing Literature Study on Test Amplification, *Journal of Systems and Software 157*, August 2019, p. 1–16, [doi:10.1016/j.jss.2019.110398], [hal:hal-02290742].

[38] J.-E. DARTOIS, J. Boukhobza, A. Knefati, O. Barais, Investigating Machine Learning Algorithms for Modeling SSD I/O Performance for Container-based Virtualization, *IEEE transactions on cloud computing 14*, 2019, p. 1–14, [doi:10.1109/TCC.2019.2898192], [hal:hal-02013421].

[39] T. Degueule, B. Combemale, A. Blouin, O. Barais, J.-M. Jézéquel, Safe Model Polymorphism for Flexible Modeling, *Computer Languages, Systems and Structures 49*, September 2016, p. 30, [doi:10.1016/j.cl.2016.09.001], [hal:hal-01367305].

[40] F. A. Fontana, G. Perrouin, A. Ampatzoglou, M. Acher, B. Walter, M. Cordy, F. Palomba, X. Devroey, MALTESQUE 2019 Workshop Summary, *Software Engineering Notes*, January 2020, [doi:10.1145/3375572.3375582], [hal:hal-02471302].

[41] I. Gonzalez-Herrera, J. Bourcier, E. Daubert, W. Rudametkin, O. Barais, F. Fouquet, J.-M. Jézéquel, B. Baudry, ScapeGoat: Spotting abnormal resource usage in component-based reconfigurable software systems, *Journal of Systems and Software*, 2016, [doi:10.1016/j.jss.2016.02.027], [hal:hal-01354999].

[42] A. Halin, A. Nuttinck, M. Acher, X. Devroey, G. Perrouin, B. Baudry, Test them all, is it worth it? Assessing configuration sampling on the JHipster Web development stack, *Empirical Software Engineering*, July 2018, p. 1–44, https://arxiv.org/abs/1710.07980, [doi:10.1007/s10664-018-9635-4], [hal:hal-01829928].

[43] A. Hervieu, D. Marijan, A. Gotlieb, B. Baudry, Optimal Minimisation of Pairwise-covering Test Configurations Using Constraint Programming, *Information and Software Technology 71*, March 2016, p. 129–146, [hal:hal-01352831].

[44] G. Kanakis, D. E. Khelladi, S. Fischer, M. Tröls, A. Egyed, An Empirical Study on the Impact of Inconsistency Feedback during Model and Code Co-changing., *The Journal of Object Technology 18*, 2, 2019, p. 10:1, [doi:10.5381/jot.2019.18.2.a10], [hal:hal-02192486].

[45] M. Kezadri, M. Pantel, X. Thirioux, B. Combemale, Correct-by-construction model driven engineering composition operators, *Formal Aspects of Computing 28*, 3, January 2016, [doi:10.1007/s00165-016-0354-6], [hal:hal-01319576].

[46] G. Khandu Narwane, J. A. Galindo Duarte, S. Narayanan Krishna, D. Benavides, J.-V. Millo, S. Ramesh, Traceability analyses between features and assets in software product lines, *Entropy 18*, 8, July 2016, p. 269, [doi:10.3390/e18080269], [hal:hal-01342351].

[47] J. Kienzle, G. Mussbacher, B. Combemale, L. Bastin, N. Bencomo, J.-M. Bruel, C. Becker, S. Betz, R. Chitchyan, B. Cheng, S. Klingert, R. Paige, B. Penzenstadler, N. Seyff, E. Syriani, C. C. Venters, Towards Model-Driven Sustainability Evaluation, *Communications of the ACM 63*, 3, 2020, p. 80–91, [doi:10.1145/3371906], [hal:hal-02146543].

[48] J. Kienzle, G. Mussbacher, B. Combemale, J. Deantoni, A Unifying Framework for Homogeneous Model Composition, *Software and Systems Modeling*, January 2019, p. 1–19, [doi:10.1007/s10270-018-00707-8], [hal:hal-01949050].

[49] R. Kretschmer, D. E. Khelladi, A. Egyed, Transforming Abstract to Concrete Repairs with a Generative Approach of Repair Values, *Journal of Systems and Software 175*, 2021, p. 19, [doi:10.1016/j.jss.2020.110889], [hal:hal-03127118].

[50] R. Kretschmer, D. E. Khelladi, R. E. Lopez-Herrejon, A. Egyed, Consistent Change Propagation within Models, *Software and Systems Modeling 20*, 2, April 2021, p. 539–555, [doi:10.1007/s10270-020-00823-4], [hal:hal-03029432].

[51] G. Le Moulec, A. Blouin, V. Gouranton, B. Arnaldi, Automatic Production of End User Documentation for DSLs, *Computer Languages, Systems and Structures 54*, July 2018, p. 337–357, Accepted for publication in COMLAN, [doi:10.1016/j.cl.2018.07.006], [hal:hal-01549042].

[52] M. Leduc, T. Degueule, E. Van Wyk, B. Combemale, The Software Language Extension Problem, *Software and Systems Modeling 19*, 2, 2020, p. 263–267, [doi:10.1007/s10270-019-00772-7], [hal:hal-02399166].

[53] M. Leduc, G. Jouneaux, T. Degueule, G. Le Guernic, O. Barais, B. Combemale, Automatic generation of Truffle-based interpreters for Domain-Specific Languages, *The Journal of Object Technology 19*, 2, 2020, p. 1–21, [doi:10.5381/jot.2020.19.2.a1], [hal:hal-02395867].

[54] B. Lelandais, M.-P. Oudot, B. Combemale, Applying Model-Driven Engineering to High-Performance Computing: Experience Report, Lessons Learned, and Remaining Challenges, *Journal of Computer Languages 55*, 2019, p. 1–10, [doi:10.1016/j.cola.2019.100919], [hal:hal-02296030].

[55] D. Leroy, E. Bousse, M. Wimmer, T. Mayerhofer, B. Combemale, W. Schwinger, Behavioral interfaces for executable DSLs, *Software and Systems Modeling 19*, 4, April 2020, p. 1015–1043, [doi:10.1007/s10270-020-00798-2], [hal:hal-02565549].

[56] D. Leroy, P. Jeanjean, E. Bousse, M. Wimmer, B. Combemale, Runtime Monitoring for Executable DSLs, *The Journal of Object Technology 19*, 2, 2020, p. 1–23, [doi:10.5381/jot.2020.19.2.a6], [hal:hal-03109992].

[57] D. Leroy, J. Sallou, J. Bourcier, B. Combemale, When Scientific Software Meets Software Engineering, *IEEE Computer Society*, 2021, p. 1–11, [hal:hal-03318348].

[58] H. Martin, M. Acher, J. A. Pereira, L. Lesoil, J.-M. Jézéquel, D. E. Khelladi, Transfer Learning Across Variants and Versions: The Case of Linux Kernel Size, *IEEE Transactions on Software Engineering*, 2021, p. 1–17, [hal:hal-03358817].

[59] D. Méndez-Acuña, J. A. Galindo Duarte, T. Degueule, B. Combemale, B. Baudry, Leveraging Software Product Lines Engineering in the Development of External DSLs: A Systematic Literature Review, *Computer Languages, Systems and Structures*, September 2016, [doi:10.1016/j.cl.2016.09.004], [hal:hal-01372702].

[60] D. A. Méndez-Acuña, J. A. Galindo, B. Combemale, A. Blouin, B. Baudry, Reverse Engineering Language Product Lines from Existing DSL Variants, *Journal of Systems and Software*, May 2017, [doi:10.1016/j.jss.2017.05.042], [hal:hal-01524632].

[61] G. Mussbacher, B. Combemale, J. Kienzle, S. Abrahão, H. Ali, N. Bencomo, M. Búr, L. Burgueño, G. Engels, P. Jeanjean, J.-M. Jézéquel, T. Kühn, S. Mosser, H. Sahraoui, E. Syriani, D. Varró, M. Weyssow, Opportunities in Intelligent Modeling Assistance, *Software and Systems Modeling 19*, 5, 2020, p. 1045–1053, [doi:10.1007/s10270-020-00814-5], [hal:hal-02876536].

[62] A. Pierantonio, M. V. D. Brand, B. Combemale, Open Access: all you wanted to know and never dared to ask, *The Journal of Object Technology 19*, 1, 2020, p. 1–4, [doi:10.5381/jot.2020.19.1.e1], [hal:hal-03137877].

[63] A. Pierantonio, M. Van Den Brand, B. Combemale, The JOT Journal: Towards a Rising Generation, *The Journal of Object Technology 18*, 1, 2019, p. 1–3, [doi:10.5381/jot.2019.18.1.e1], [hal:hal-02408017].

[64] R. Potvin, J. Levenberg, Why Google stores billions of lines of code in a single repository, *Communications of the ACM 59*, 7, 2016, p. 78–87.

[65] I. Tariq Javed, R. Copeland, N. Crespi, M. Emmelmann, A. A. Corici, A. Bouabdallah, T. Zhang, S. El Jaouhari, F. Beierle, S. Göndör, A. Küpper, K. Corre, J.-M. Crom, F. Oberle, I. Friese, A. Caldeira, G. Dias, N. Santos, R. Chaves, R. Lopes Pereira, Cross-domain identity and discovery framework for web calling services, *Annals of Telecommunications - annales des télécommunications 72*, 7-8, August 2017, p. 459 – 468, [doi:10.1007/s12243-017-0587-2], [hal:hal-01596116].

[66] P. Temple, M. Acher, J.-M. Jézéquel, O. Barais, Learning-Contextual Variability Models, *IEEE Software 34*, 6, November 2017, p. 64–70, [doi:10.1109/MS.2017.4121211], [hal:hal-01659137].

[67] P. Temple, M. Acher, J.-M. Jézéquel, Empirical Assessment of Multimorphic Testing, *IEEE Transactions on Software Engineering*, July 2019, p. 1–21, [doi:10.1109/TSE.2019.2926971], [hal:hal-02177158].

[68] P. Temple, G. Perrouin, M. Acher, B. Biggio, J.-M. Jézéquel, F. Roli, Empirical Assessment of Generating Adversarial Configurations for Software Product Lines, *Empirical Software Engineering 26*, 6, December 2020, p. 1–57, [doi:10.1007/s10664-020-09915-7], [hal:hal-03045797].

[69] X. Tërnava, P. Collet, A framework for managing the imperfect modularity of variability implementations, *Journal of Computer Languages*, September 2020, p. 1–39, [doi:10.1016/j.cola.2020.100998], [hal:hal-02951745].

[70] O. L. Vera-Pérez, B. Danglot, M. Monperrus, B. Baudry, A Comprehensive Study of Pseudo-tested Methods, *Empirical Software Engineering*, 2018, p. 1–33, [doi:10.1007/s10664-018-9653-2], [hal:hal-01867423].

[71] A. Wortmann, O. Barais, B. Combemale, M. Wimmer, Modeling Languages in Industry 4.0: An Extended Systematic Mapping Study, *Software and Systems Modeling 19*, 1, 2020, p. 67–94, [doi:10.1007/s10270-019-00757-6], [hal:hal-02282028].

[72] J. Xuan, B. Cornu, M. Martinez, B. Baudry, L. Seinturier, M. Monperrus, B-Refactoring: Automatic Test Code Refactoring to Improve Dynamic Analysis, *Information and Software Technology 76*, 2016, p. 65–80, [doi:10.1016/j.infsof.2016.04.016], [hal:hal-01309004].

[73] K. Yeboah-Antwi, B. Baudry, Online Genetic Improvement on the java virtual machine with ECSELR, *Genetic Programming and Evolvable Machines*, October 2016, p. 1–27, [doi:10.1007/s10710-016-9278-4], [hal:hal-01382964].

Articles in National Journals

[74] J.-M. Jézéquel, J. Jongwane, Comment maîtriser la complexité des logiciels ?, *Interstices*, February 2017, [hal:hal-01503821].

Invited Conferences

[75] B. Combemale, Bringing Intelligence to Sociotechnical IoT Systems: Modeling Opportunities and Challenges, *in: MDE4IoT 2019 - 3nd International Workshop on Model-Driven Engineering for the Internet-of-Things*, p. 1–2, Munich, Germany, September 2019, [hal:hal-02285737].

[76] G. Perrouin, M. Acher, J.-M. Davril, A. Legay, P. Heymans, A Complexity Tale: Web Configurators, *in: VACE 2016 - 1st International Workshop on Variability and Complexity in Software Design Pages (co-located with ICSE'16)*, p. 28 – 31, Austin, United States, May 2016, [doi:10.1145/2897045.2897051], [hal:hal-01427165].

International Conferences with Proceedings

[77] M. Acher, R. E. Lopez-Herrejon, R. Rabiser, Teaching Software Product Lines: A Snapshot of Current Practices and Challenges (Journal-First Abstract), *in: SPLC2018 - 22nd International Systems and Software Product Line Conference*, p. 1, Gothenburg, Sweden, September 2018, [hal:hal-01829933].

[78] M. Acher, P. Temple, J.-M. Jézéquel, J. Á. Galindo Duarte, J. Martinez, T. Ziadi, VaryLaTeX: Learning Paper Variants That Meet Constraints, *in: VaMoS 2018 - 12th International Workshop on Variability Modelling of Software-Intensive Systems*, ACM, p. 83–88, Madrid, Spain, February 2018, [doi:10.1145/3168365.3168372], [hal:hal-01659161].

[79] M. Acher, T. Ziadi, R. E. Lopez-Herrejon, J. Martinez, Seventh international workshop on reverse variability engineering (REVE 2019), *in: SPLC 2019 - 23rd International Systems and Software Product Line Conference*, ACM Press, p. 1, Paris, France, September 2019, [hal:hal-02268373].

[80] M. Acher, Learning the Linux Kernel Configuration Space: Results and Challenges, *in: ELC Europe 2019 - Embedded Linux Conference Europe 2019*, p. 1–49, Lyon, France, October 2019, [hal:hal-02342130].

[81] A. Alidra, A. Beugnard, H. Godfroy, P. Kimmel, G. Le Guernic, A Language Agnostic Approach to Modeling Requirements: Specification and Verification, *in: MODELS '20 Companion*, Virtual Event, Canada, October 2020, [doi:10.1145/3417990.3419224], [hal:hal-02924645].

[82] A. Allen, C. Aragon, C. Becker, J. C. Carver, A. Chis, B. Combemale, M. Croucher, K. CROWSTON, D. Garijo, A. Gehani, C. Goble, R. Haines, R. Hirschfeld, J. Howison, K. Huff, C. Jay, D. Katz, C. Kirchner, K. Kuksenok, R. Lämmel, O. Nierstrasz, M. Turk, R. V. van Nieuwpoort, M. Vaughn, J. Vinju, Lightning Talk:"I solemnly pledge" A Manifesto for Personal Responsibility in the Engineering of Academic Software, *in: Fourth Workshop on Sustainable Software for Science: Practice and Experiences (WSSSPE4)*, G. Allen, J. Carver, M. Heroux, L. J. Hwang, D. S. Katz, K. E. Niemeyer, M. Parashar, C. C. Venters, S.-C. T. Choi, T. Crick, M. R. Crusoe, S. Gesing, R. Haines (editors), *Proceedings of the Fourth Workshop on Sustainable Software for Science: Practice and Experiences (WSSSPE4)*, *1686*, CEUR, Manchester, United Kingdom, 2016, [hal:hal-01367344].

[83] J. Alves Pereira, M. Acher, H. Martin, J.-M. Jézéquel, Sampling Effect on Performance Prediction of Configurable Systems: A Case Study, *in: ICPE 2020 - 11th ACM/SPEC International Conference on Performance Engineering*, ACM, ACM, p. 1–13, Edmonton, Canada, April 2020, [hal:hal-02356290].

[84] B. Amand, M. Cordy, P. Heymans, M. Acher, P. Temple, J.-M. Jézéquel, Towards Learning-Aided Configuration in 3D Printing: Feasibility Study and Application to Defect Prediction, *in: VAMOS 2019 - 13th International Workshop on Variability Modelling of Software-Intensive Systems*, ACM, p. 1–9, Leuven, Belgium, February 2019, [doi:10.1145/3302333.3302338], [hal:hal-01990767].

[85] M. Bagherzadeh, K. Jahed, B. Combemale, J. Dingel, Live-UMLRT: A Tool for Live Modeling of UML-RT Models, *in: MODELS 2019 - ACM/IEEE 22nd International Conference on Model Driven Engineering Languages and Systems*, IEEE, p. 743–747, Munich, Germany, September 2019, [doi:10.1109/MODELS-C.2019.00115], [hal:hal-02407932].

[86] O. Barais, J. Bourcier, Y.-D. Bromberg, C. Dion, Towards microservices architecture to transcode videos in the large at low costs, *in: TEMU 2016 - International Conference on Telecommunications and Multimedia*, p. 1 – 6, Heraklion, Greece, July 2016, [doi:10.1109/TEMU.2016.7551918], [hal:hal-01427277].

[87] O. Barais, B. Combemale, A. Wortmann, Language Engineering with the GEMOC Studio, *in: ICSAW 2017 - IEEE International Conference on Software Architecture Workshops*, IEEE (editor), p. 3, Gothenburg, Sweden, April 2017, [hal:hal-01609576].

[88] R. Belafia, P. Jeanjean, O. Barais, G. Le Guernic, B. Combemale, From Monolithic to Microservice Architecture: The Case of Extensible and Domain-Specific IDEs, *in: MODELS 2021: ACM/IEEE 24th International Conference on Model Driven Engineering Languages and Systems*, IEEE, p. 1–10, Virtual, Japan, October 2021, [hal:hal-03342678].

[89] A. Benelallam, N. Harrand, C. Soto-Valero, B. Baudry, O. Barais, The Maven Dependency Graph: a Temporal Graph-based Representation of Maven Central, *in: MSR 2019 - 16th International Conference on Mining Software Repositories*, ACM, p. 344–348, Montreal, Canada, May 2019, [doi:10.1109/MSR.2019.00060], [hal:hal-02080243].

[90] A. Benelallam, T. Hartmann, L. Mouline, F. FOUQUET, J. Bourcier, O. Barais, Y. Le Traon, Raising Time Awareness in Model-Driven Engineering, *in: ACM/IEEE 20th International Conference on Model Driven Engineering Languages and Systems.*, Austin, Texas, United States, September 2017, [hal:hal-01580554].

[91] F. Bordeleau, B. Combemale, R. Eramo, M. V. D. Brand, M. Wimmer, Tool-Support of Socio-Technical Coordination in the Context of Heterogeneous Modeling: A Research Statement and Associated Roadmap, *in: GEMOC 2018 - 6th International Workshop on The Globalization of Modeling Languages*, p. 1–3, Copenhagen, Denmark, October 2018, [hal:hal-01958443].

[92] F. Bordeleau, B. Combemale, R. Eramo, M. Van Den Brand, M. Wimmer, Towards Model-Driven Digital Twin Engineering: Current Opportunities and Future Challenges, *in: ICSMM 2020 - International Conference on Systems Modelling and Management*, Bergen, Norway, June 2020, [hal:hal-02946949].

[93] M. Boussaa, O. Barais, B. Baudry, G. Sunyé, Automatic Non-functional Testing of Code Generators Families, *in: 15th International Conference on Generative Programming: Concepts & Experiences (GPCE 2016)*, *Proceedings of the 2016 ACM SIGPLAN International Conference on Generative Programming: Concepts and Experiences*, p. 11, Amsterdam, Netherlands, October 2016, [hal:hal-01356849].

[94] M. Boussaa, O. Barais, B. Baudry, G. Sunyé, NOTICE: A Framework for Non-functional Testing of Compilers, *in: 2016 IEEE International Conference on Software Quality, Reliability & Security (QRS 2016)*, *IEEE Computer Society Conference Publishing Services (CPS)*, Vienna, Austria, August 2016, [hal:hal-01344835].

[95] E. Bousse, T. Degueule, D. Vojtisek, T. Mayerhofer, J. Deantoni, B. Combemale, Execution Framework of the GEMOC Studio (Tool Demo), *in: Proceedings of the 2016 ACM SIGPLAN International Conference on Software Language Engineering*, *SLE 2016*, p. 8, Amsterdam, Netherlands, October 2016, [hal:hal-01355391].

[96] J.-M. Bruel, B. Combemale, E. Guerra, J.-M. Jézéquel, J. Kienzle, J. De Lara, G. Mussbacher, E. Syriani, H. Vangheluwe, Model Transformation Reuse across Metamodels - A classification and comparison of approaches, *in: ICMT 2018 - International Conference on Theory and Practice of Model Transformations*, *LNCS*, *10888*, Springer, p. 92–109, Toulouse, France, June 2018, [doi:10.1007/978-3-319-93317-7_4], [hal:hal-01910113].

[97] A. Cheron, J. Bourcier, O. Barais, A. Michel, Comparison Matrices of Semantic RESTful APIs Technologies, *in: ICWE 2019 - 19th International Conference On Web Engineering*, *LNCS*, *11496*, Springer, p. 425–440, Daejeon, South Korea, June 2019, [doi:10.1007/978-3-030-19274-7_30], [hal:hal-02114296].

[98] 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, [hal:hal-01258358].

[99] B. Combemale, B. H. Cheng, A. Moreira, J.-M. Bruel, J. Gray, Modeling for Sustainability, *in: Modeling in Software Engineering 2016 (MiSE'16)*, *8th IEEE/ACM International Workshop on Modeling in Software Engineering (MiSE 2016)*, ACM, p. 62–66, Austin, United States, May 2016, [hal:hal-01517387].

[100] F. Coulon, A. Auvolat, B. Combemale, Y.-D. Bromberg, F. Taïani, O. Barais, N. Plouzeau, Modular and Distributed IDE, *in: SLE 2020 - 13th ACM SIGPLAN International Conference on Software Language Engineering*, p. 270–282, Virtual, United States, November 2020, [doi:10.1145/3426425.3426947], [hal:hal-02964806].

[101] F. Coulon, T. Degueule, T. Van Der Storm, B. Combemale, Shape-Diverse DSLs: Languages without Borders (Vision Paper), *in: SLE 2018 - 11th ACM SGIPLAN International Conference on Software Language Engineering*, ACM, p. 215–219, Boston, United States, November 2018, [doi:10.1145/3276604.3276623], [hal:hal-01889155].

[102] J.-E. DARTOIS, J. Boukhobza, V. Francoise, O. Barais, Tracking Application Fingerprint in a Trustless Cloud Environment for Sabotage Detection, *in: MASCOTS 2019 - 27th IEEE International Symposium on the Modeling, Analysis, and Simulation of Computer and Telecommunication Systems*, IEEE, p. 74–82, Rennes, France, October 2019, [doi:10.1109/MASCOTS.2019.00018], [hal:hal-02303153].

[103] J.-E. Dartois, A. Knefati, J. Boukhobza, O. Barais, Using Quantile Regression for Reclaiming Unused Cloud Resources while achieving SLA, *in: CloudCom 2018 - 10th IEEE International Conference on Cloud Computing Technology and Science*, IEEE, p. 89–98, Nicosia, Cyprus, December 2018, [doi:10.1109/CloudCom2018.2018.00030], [hal:hal-01898438].

[104] J.-E. DARTOIS, I. Meriau, M. Handaoui, J. Boukhobza, O. Barais, Leveraging cloud unused resources for Big data application while achieving SLA, *in: MASCOTS 2019 - 27th IEEE International Symposium on the Modeling, Analysis, and Simulation of Computer and Telecommunication Systems*, IEEE, p. 1–2, Rennes, France, October 2019, [hal:hal-02362257].

[105] J.-E. DARTOIS, H. B. Ribeiro, J. Boukhobza, O. Barais, Cuckoo: Opportunistic MapReduce on Ephemeral and Heterogeneous Cloud Resources, *in: CLOUD 2019 - IEEE 12th International Conference on Cloud Computing*, IEEE, p. 1–8, Milan, Italy, July 2019, [doi:10.1109/CLOUD.2019.00070], [hal:hal-02179453].

[106] J. A. Galindo Duarte, M. Acher, J. M. Tirado, C. Vidal, B. Baudry, D. Benavides, Exploiting the Enumeration of All Feature Model Configurations : A New Perspective with Distributed Computing, *in: Software Product Line Conference*, Beijing, China, September 2016, [hal:hal-01334851].

[107] A. Gómez-Boix, D. Frey, Y.-D. Bromberg, B. Baudry, A Collaborative Strategy for mitigating Tracking through Browser Fingerprinting, *in: MTD 2019 - 6th ACM Workshop on Moving Target Defense*, p. 1–12, London, United Kingdom, November 2019, [doi:10.1145/3338468.3356828], [hal:hal-02282591].

[108] A. Gómez-Boix, P. Laperdrix, B. Baudry, Fingerprinting mobile devices: A short analysis, *in: CIEL 2017 - 6ème Conférence en IngénieriE du Logiciel*, Montpellier, France, June 2017, [hal:hal-01611101].

[109] A. Gómez-Boix, P. Laperdrix, B. Baudry, Hiding in the Crowd: an Analysis of the Effectiveness of Browser Fingerprinting at Large Scale, *in: WWW2018 - TheWebConf 2018 : 27th International World Wide Web Conference*, p. 1–10, Lyon, France, April 2018, [doi:10.1145/3178876.3186097], [hal:hal-01718234].

[110] I. Gonzalez-Herrera, J. Bourcier, W. Rudametkin, O. Barais, F. Fouquet, Squirrel: Architecture Driven Resource Management, *in: SAC - 31st Annual ACM Symposium on Applied Computing*, Pisa, Italy, April 2016, [doi:10.1145/0000000.0000000], [hal:hal-01355000].

[111] A. Halin, A. Nuttinck, M. Acher, X. Devroey, G. Perrouin, P. Heymans, Yo Variability! JHipster: A Playground for Web-Apps Analyses, *in: 11th International Workshop on Variability Modelling of Software-intensive Systems*, p. 44 – 51, Eindhoven, Netherlands, February 2017, [doi:10.1145/3023956.3023963], [hal:hal-01468084].

[112] M. Handaoui, J.-E. Dartois, J. Boukhobza, O. Barais, L. D'orazio, ReLeaSER: A Reinforcement Learning Strategy for Optimizing Utilization Of Ephemeral Cloud Resources, *in: CloudCom 2020 - 12th IEEE International Conference on Cloud Computing Technology and Science*, IEEE, p. 1–9, Bangkok, Thailand, December 2020, [hal:hal-02989286].

[113] M. Handaoui, J.-E. Dartois, L. Lemarchand, J. Boukhobza, Salamander: a Holistic Scheduling of MapReduce Jobs on Ephemeral Cloud Resources, *in: CCGRID 2020 - 20th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*, IEEE, p. 1–10, Melbourne, Australia, November 2020, [hal:hal-02497029].

[114] M. Heinz, R. Lämmel, M. Acher, Discovering Indicators for Classifying Wikipedia Articles in a Domain: A Case Study on Software Languages, *in: SEKE 2019 - The 31st International Conference on Software Engineering and Knowledge Engineering*, p. 1–6, Lisbonne, Portugal, July 2019, [hal:hal-02129131].

[115] A. Jangda, M. Mishra, B. Baudry, libmask: Protecting Browser JIT Engines from the Devil in the Constants, *in: Annual Conference on Privacy, Security and Trust*, Auckland, New Zealand, December 2016, [hal:hal-01382971].

[116] P. Jeanjean, B. Combemale, O. Barais, From DSL Specification to Interactive Computer Programming Environment, *in: SLE 2019 - 12th ACM SIGPLAN International Conference on Software Language Engineering*, ACM, p. 167–178, Athènes, Greece, October 2019, [doi:10.1145/3357766.3359540], [hal:hal-02307953].

[117] P. Jeanjean, B. Combemale, O. Barais, IDE as Code: Reifying Language Protocols as First-Class Citizens, *in: ISEC 2021 - Innovations in Software Engineering Conference*, p. 1–5, Bhubaneswar / Virtual, India, February 2021, [hal:hal-03107122].

[118] G. Jouneaux, O. Barais, B. Combemale, G. Mussbacher, SEALS: A framework for building Self-Adaptive Virtual Machines, *in: SLE 2021 - 14th ACM SIGPLAN International Conference on Software Language Engineering*, ACM, p. 1–14, Chicago, United States, October 2021, [doi:10.1145/3486608.3486912], [hal:hal-03355253].

[119] G. Jouneaux, O. Barais, B. Combemale, G. Mussbacher, Towards Self-Adaptable Languages, *in: Onward! 2021 - ACM SIGPLAN International Symposium on New Ideas, New Paradigms, and Reflections on Programming and Software*, ACM, p. 1–16, Chicago, United States, October 2021, [hal:hal-03318816].

[120] G. Kanakis, S. Fischer, D. E. Khelladi, A. Egyed, Supporting A Flexible Grouping Mechanism for Collaborating Engineering Teams, *in: ICGSE 2019 - 14th ACM/IEEE International Conference on Global Software Engineering*, IEEE, p. 129–138, Montreal, QC, Canada, May 2019, [doi:10.1109/ICGSE.2019.00033], [hal:hal-02192482].

[121] D. E. Khelladi, B. Combemale, M. Acher, O. Barais, J.-M. Jézéquel, Co-Evolving Code with Evolving Metamodels, *in: ICSE 2020 - 42nd International Conference on Software Engineering*, p. 1–13, Séoul, South Korea, July 2020, [hal:hal-03029429].

[122] D. E. Khelladi, B. Combemale, M. Acher, O. Barais, On the Power of Abstraction: a Model-Driven Co-evolution Approach of Software Code, *in: ICSE-NIER 2020 - ACM/IEEE 42nd International Conference on Software Engineering, New Ideas and Emerging Results*, ACM, p. 1–4, Séoul, South Korea, October 2020, [hal:hal-03029426].

[123] D. E. Khelladi, R. Kretschmer, A. Egyed, Detecting and Exploring Side Effects when Repairing Model Inconsistencies, *in: SLE 2019 - 12th ACM SIGPLAN International Conference on Software Language Engineering*, ACM, p. 103–126, Athènes, Greece, October 2019, [doi:10.1145/3357766.3359546], [hal:hal-02326034].

[124] J. Kienzle, G. Mussbacher, O. Alam, M. Schöttle, N. Belloir, P. Collet, B. Combemale, J. Deantoni, J. Klein, B. Rumpe, VCU: The Three Dimensions of Reuse, *in: 15th International Conference on Software Reuse (ICSR-15), The 15th International Conference on Software Reuse*, Limassol, Cyprus, May 2016, [hal:hal-01287720].

[125] 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, *in: Symposium on Applied Computing, Proceedings of the 31st Annual ACM Symposium on Applied Computing*, ACM, p. 1562 – 1565, Pise, Italy, April 2016, [doi:10.1145/2851613.2851977], [hal:hal-01241673].

[126] P. Laperdrix, B. Baudry, V. Mishra, FPRandom: Randomizing core browser objects to break advanced device fingerprinting techniques, *in: ESSoS 2017 - 9th International Symposium on Engineering Secure Software and Systems*, p. 17, Bonn, Germany, July 2017, [hal:hal-01527580].

[127] P. Laperdrix, W. Rudametkin, B. Baudry, Beauty and the Beast: Diverting modern web browsers to build unique browser fingerprints, *in: 37th IEEE Symposium on Security and Privacy (S&P 2016)*, San Jose, United States, May 2016, [hal:hal-01285470].

[128] Q. Le Dilavrec, D. E. Khelladi, A. Blouin, J.-M. Jézéquel, Untangling Spaghetti of Evolutions in Software Histories to Identify Code and Test Co-evolutions, *in: ICMSE'21 - 37th International Conference on Software Maintenance and Evolution*, p. 1–11, Virtual Event, Luxembourg, September 2021, [hal:hal-03340174].

[129] G. Le Guernic, B. Combemale, J. A. Galindo Duarte, Industrial Experience Report on the Formal Specification of a Packet Filtering Language Using the K Framework, *in: 3rd Workshop on Formal Integrated Development Environment*, Catherine Dubois and Dominique Mery and Paolo Masci, Limassol, Cyprus, November 2016, [hal:hal-01401849].

[130] G. Le Guernic, Experience Report on the Development of a Specialized Multi-view Multi-stakeholder Model-Based Engineering Framework, *in: DSM 2019 - 17th ACM SIGPLAN International Workshop on Domain-Specific Modeling*, ACM Press, p. 50–59, Athens, Greece, October 2019, [doi:10.1145/3358501.3361237], [hal:hal-02398053].

[131] G. Le Moulec, F. Argelaguet Sanz, V. Gouranton, A. Blouin, B. Arnaldi, AGENT: Automatic Generation of Experimental Protocol Runtime, *in: ACM Symposium on Virtual Reality Software and Technology (VRST), Virtual Reality Software and Technology*, Gothenburg, Sweden, November 2017, [hal:hal-01613873].

[132] J. Le Noir, S. Madelénat, C. Labreuche, O. Constant, G. Gailliard, M. Acher, O. Barais, A Decision-making Process for Exploring Architectural Variants in Systems Engineering, *in: Software Product Lines Conference (SPLC)*, Beijing, China, September 2016, [doi:10.1145/2934466.2946045], [hal:hal-01374140].

[133] M. Leduc, T. Degueule, B. Combemale, T. Van Der Storm, O. Barais, Revisiting Visitors for Modular Extension of Executable DSMLs, *in: 2017 ACM/IEEE 20th International Conference on Model Driven Engineering Languages and Systems*, Austin, United States, September 2017, [doi:10.1109/MODELS.2017.23], [hal:hal-01568169].

[134] M. Leduc, T. Degueule, B. Combemale, Modular Language Composition for the Masses, *in: SLE 2018 - 11th ACM SIGPLAN International Conference on Software Language Engineering*, p. 1–12, Boston, United States, November 2018, [doi:10.1145/3276604.3276622], [hal:hal-01890446].

[135] B. Lelandais, M.-P. Oudot, B. Combemale, Fostering metamodels and grammars within a dedicated environment for HPC: the NabLab environment (tool demo), *in: SLE 2018 - International Conference on Software Language Engineering*, p. 1–9, Boston, United States, November 2018, [doi:10.1145/3276604.3276620], [hal:hal-01910139].

[136] V. Lelli, A. Blouin, B. Baudry, F. Coulon, O. Beaudoux, Automatic Detection of GUI Design Smells: The Case of Blob Listener, *in: Proceedings of the 8th ACM SIGCHI Symposium on Engineering Interactive Computing Systems (EICS'16)*, p. 263–274, Brussels, Belgium, June 2016, [doi:10.1145/2933242.2933260], [hal:hal-01308625].

[137] D. Leroy, E. Bousse, A. Megna, B. Combemale, M. Wimmer, Trace Comprehension Operators for Executable DSLs, *in: ECMFA 2018 - 14th European Conference on Modelling Foundations and Applications*, *LNCS*, *10890*, Springer, p. 293–310, Toulouse, France, June 2018, [doi:10.1007/978-3-319-92997-2_19], [hal:hal-01803031].

[138] D. Leroy, E. Bousse, M. Wimmer, B. Combemale, W. Schwinger, Create and Play your Pac-Man Game with the GEMOC Studio (Tool Demonstration), *in: EXE 2017 - 3rd International Workshop on Executable Modeling*, p. 1–6, Austin, United States, September 2017, [hal:hal-01651801].

[139] D. Leroy, B. Lelandais, M.-P. Oudot, B. Combemale, Monilogging for Executable Domain-Specific Languages, *in: SLE 2021 - 13th International Conference on Software Language Engineering*, ACM, p. 1–14, Chicago, United States, October 2021, [doi:10.1145/3486608.3486906], [hal:hal-03358061].

[140] L. Lesoil, M. Acher, A. Blouin, J.-M. Jézéquel, Deep Software Variability: Towards Handling Cross-Layer Configuration, *in: VaMoS 2021 - 15th International Working Conference on Variability Modelling of Software-Intensive Systems*, p. 1–8, Krems / Virtual, Austria, February 2021, [hal:hal-03084276].

[141] L. Lesoil, M. Acher, X. Tërnava, A. Blouin, J.-M. Jézéquel, The Interplay of Compile-time and Run-time Options for Performance Prediction, *in: SPLC 2021 - 25th ACM International Systems and Software Product Line Conference - Volume A*, ACM, p. 1–12, Leicester, United Kingdom, September 2021, [doi:10.1145/3461001.3471149], [hal:hal-03286127].

[142] G. Lyan, D. Gross-Amblard, J.-M. Jézéquel, On the Quality of Compositional Prediction for Prospective Analytics on Graphs, *in: The 23rd International Conference on Big Data Analytics and Knowledge Discovery*, p. 91–105, Linz, Austria, September 2021, [doi:10.1007/978-3-030-87101-7_10], [hal:hal-03356199].

[143] G. Lyan, J.-M. Jézéquel, D. Gross-Amblard, B. Combemale, DataTime: a Framework to smoothly Integrate Past, Present and Future into Models, *in: MODELS 2021 - ACM/IEEE 24th International Conference on Model Driven Engineering Languages and Systems*, p. 1–11, Fukuoka, Japan, October 2021, [hal:hal-03355162].

[144] H. Martin, M. Acher, J. A. Pereira, J.-M. Jézéquel, A comparison of performance specialization learning for configurable systems, *in: SPLC 2021 - 25th ACM International Systems and Software Product Line Conference*, ACM, p. 1–11, Leicester, United Kingdom, September 2021, [doi:10.1145/3461001.3471155], [hal:hal-03335263].

[145] H. Martin, J. A. Pereira, M. Acher, P. Temple, Machine Learning and Configurable Systems: A Gentle Introduction, *in: SPLC 2019 - 23rd International Systems and Software Product Line Conference*, ACM, p. 83–88, Paris, France, September 2019, [doi:10.1145/3336294.3342383], [hal:hal-02287459].

[146] J. Martinez, J.-S. Sottet, A. García Frey, T. Bissyandé, T. Ziadi, J. Klein, P. Temple, M. Acher, Y. Le Traon, Towards Estimating and Predicting User Perception on Software Product Variants, *in: ICSR 2018 - International Conference on Software Reuse*, *LNCS*, *10826*, Springer, p. 23–40, Madrid, Spain, May 2018, [doi:10.1007/978-3-319-90421-4_2], [hal:hal-01720519].

[147] D. Méndez-Acuña, J. Galindo, B. Combemale, A. Blouin, B. Baudry, Reverse engineering language product lines from existing DSL variants, *in: 22nd International Systems and Software Product Line Conference - SPLC 2018*, *1*, ACM : Association for Computing Machinery, p. 148, Gothenburg, Sweden, September 2018, [doi:10.1145/3233027.3236394], [hal:hal-02982981].

[148] D. Méndez-Acuña, J. A. Galindo Duarte, B. Combemale, A. Blouin, B. Baudry, G. Le Guernic, Reverse-engineering reusable language modules from legacy domain-specific languages, *in: International Conference on Software Reuse*, *Proceedings of the International Conference on Software Reuse*, Limassol, Cyprus, June 2016, [hal:hal-01284816].

[149] D. Méndez-Acuña, J. A. Galindo Duarte, B. Combemale, A. Blouin, B. Baudry, Puzzle: A tool for analyzing and extracting specification clones in DSLs, *in: ICSR 2016 the 15th International Conference on Software Reuse*, *Proceedings of the 15th International Conference on Software Reuse*, Limassol, Cyprus, June 2016, [hal:hal-01284822].

[150] N. Messe, N. Belloir, V. Chiprianov, I. Cherfa, R. Fleurquin, S. Sadou, Development of Secure System of Systems Needing a Rapid Deployment, *in: 2019 14th Annual Conference System of Systems Engineering (SoSE)*, IEEE (editor), Association for the Scientific Study of Consciousness, Anchorage, United States, May 2019, [doi:10.1109/SYSOSE.2019.8753857], [hal:hal-02948867].

[151] N. Messe, N. Belloir, V. Chiprianov, J. El-Hachem, R. Fleurquin, S. Sadou, An Asset-Based Assistance for Secure by Design, *in: APSEC 2020 - 27th Asia-Pacific Software Engineering Conference*, p. 1–10, Singapore, Singapore, December 2020, [hal:hal-02990897].

[152] N. Messe, V. Chiprianov, N. Belloir, J. El-Hachem, R. Fleurquin, S. Sadou, Asset-Oriented Threat Modeling, *in: TrustCom 2020 - 19th IEEE International Conference on Trust, Security and Privacy in Computing and Communications*, IEEE, p. 1–11, Guangzhou, China, December 2020, [hal:hal-02990919].

[153] B. Morin, F. Fleurey, K. E. Husa, O. Barais, A Generative Middleware for Heterogeneous and Distributed Services, *in: 19th International ACM Sigsoft Symposium on Component-Based Software Engineering (CBSE 2016)*, *19th International ACM Sigsoft Symposium on Component-Based Software Engineering*, Venise, Italy, April 2016, [doi:10.1109/CBSE.2016.12], [hal:hal-01356104].

[154] J. Mortara, X. Tërnava, P. Collet, A.-M. Dery-Pinna, Extending the Identification of Object-Oriented Variability Implementations using Usage Relationships, *in: SPLC 2021 - 25th ACM International Systems and Software Product Line Conference*, *Volume B*, ACM, p. 1–8, Leicester, United Kingdom, September 2021, [doi:10.1145/3461002.3473943], [hal:hal-03284626].

[155] L. Mouline, A. Benelallam, F. Fouquet, J. Bourcier, O. Barais, A Temporal Model for Interactive Diagnosis of Adaptive Systems, *in: ICAC 2018 - IEEE International Conference on Autonomic Computing*, p. 1–6, Trento, Italy, September 2018, [hal:hal-01862964].

[156] L. Mouline, A. Benelallam, T. Hartmann, F. Fouquet, J. Bourcier, B. Morin, O. Barais, Enabling Temporal-Aware Contexts for Adaptative Distributed Systems Temporal Context Representation System updates, *in: SAC 2018 - The 33rd ACM/SIGAPP Symposium On Applied Computing*, *SAC 2018: SAC 2018: Symposium on Applied Computing , April 9–13, 2018, Pau, France*, p. 1–8, Pau, France, April 2018, [doi:10.1145/3167132.3167286], [hal:hal-01723451].

[157] L. Mouline, T. Hartmann, F. Fouquet, Y. Le Traon, J. Bourcier, O. Barais, Weaving Rules into Models@run.time for Embedded Smart Systems, *in: Programming 2017 - Companion to the first International Conference on the Art, Science and Engineering of Programming*, p. 1 – 6, Brussels, Belgium, April 2017, [doi:10.1145/3079368.3079394], [hal:hal-01609796].

[158] G. Mussbacher, B. Combemale, S. Abrahão, N. Bencomo, L. Burgueño, G. Engels, J. Kienzle, T. Kühn, S. Mosser, H. Sahraoui, M. Weyssow, Towards an Assessment Grid for Intelligent Modeling Assistance, *in: MDE Intelligence 2020 - 2nd Workshop on Artificial Intelligence and Model-driven Engineering*, *10*, p. 1–10, Montreal, Canada, October 2020, [hal:hal-02925142].

[159] Y. Ndiaye, N. Aillery, O. Barais, A. Blouin, A. Bouabdallah, Modélisation et Évaluation de la Sécurité des IHM, *in: CIEL 2017 : 6ème Conférence en IngénieriE du Logiciel*, Montpellier, France, June 2017, [hal:hal-01611324].

[160] Y. Ndiaye, O. Barais, A. Blouin, A. Bouabdallah, N. Aillery, Requirements for preventing logic flaws in the authentication procedure of web applications, *in: SAC 2019 - 34th ACM/SIGAPP Symposium On Applied Computing*, p. 1–9, Limassol, Cyprus, April 2019, [doi:10.1145/3297280.3297438], [hal:hal-02087663].

[161] L. Noureddine, A. Heuser, C. Puodzius, O. Zendra, SE-PAC: A Self-Evolving PAcker Classifier against rapid packers evolution, *in: CODASPY '21 - 11th ACM Conference on Data and Application Security and Privacy*, ACM, p. 1–12, Virtual Event, United States, April 2021, [doi:10.1145/3422337.3447848], [hal:hal-03149211].

[162] E. Outin, J.-E. Dartois, O. Barais, J.-L. Pazat, Seeking for the Optimal Energy Modelisation Accuracy to Allow Efficient Datacenter Optimizations, *in: 16th International Symposium on Cluster, Cloud and Grid Computing*, *IEEE/ACM 16th International Symposium on Cluster, Cloud and Grid Computing*, Cartagena, Italy, May 2016, [doi:10.1109/CCGrid.2016.67], [hal:hal-01356099].

[163] J. A. Pereira, H. Martin, P. Temple, M. Acher, Machine Learning and Configurable Systems: A Gentle Introduction, *in: SPLC 2020 - 24th ACM International Systems and Software Product Line Conference*, p. 1, Montreal, Canada, October 2020, [doi:10.1145/3382025.3414976], [hal:hal-03020125].

[164] G. Perrouin, M. Amrani, M. Acher, B. Combemale, A. Legay, P.-Y. Schobbens, Featured model types: Towards Systematic Reuse in Modelling Language Engineering, *in: MiSE '16 - 8th International Workshop on Modeling in Software Engineering*, ACM, p. 1 – 7, New york, United States, May 2016, [doi:10.1145/2896982.2896987], [hal:hal-01406507].

[165] Q. Plazar, M. Acher, S. Bardin, A. Gotlieb, Efficient and Complete FD-Solving for Extended Array Constraints *, *in: IJCAI 2017*, Melbourne, Australia, August 2017, [hal:hal-01545557].

[166] Q. Plazar, M. Acher, G. Perrouin, X. Devroey, M. Cordy, Uniform Sampling of SAT Solutions for Configurable Systems: Are We There Yet?, *in: ICST 2019 - 12th IEEE International Conference on Software Testing, Verification, and Validation*, IEEE, p. 240–251, Xian, China, April 2019, [doi:10.1109/ICST.2019.00032], [hal:hal-01991857].

[167] A. A. Pranata, O. Barais, J. Bourcier, L. Noirie, Misconfiguration Discovery with Principal Component Analysis for Cloud-Native Services, *in: UCC 2020 - 13th IEEE/ACM International Conference on Utility and Cloud Computing*, p. 269–278, Leicester / Virtual, United Kingdom, December 2020, [hal:hal-03137874].

[168] C. Puodzius, O. Zendra, A. Heuser, L. Noureddine, Accurate and Robust Malware Analysis through Similarity of External Calls Dependency Graphs (ECDG), *in: ARES 2021 - The 16th International Conference on Availability, Reliability and Security*, p. 1–12, Virtual, Austria, August 2021. Best Paper Award IWCC 2021., [doi:10.1145/3465481.3470115], [hal:hal-03328395].

[169] A. Rio, Y. Maurel, O. Barais, Y. Bugni, Efficient use of local energy: An activity oriented modeling to guide Demand Side Management, *in: MODELS 2018 - 21th ACM/IEEE International Conference on Model Driven Engineering Languages and Systems*, ACM Press, p. 458–468, Copenhagen, Denmark, October 2018, [doi:10.1145/3239372.3239391], [hal:hal-01913169].

[170] A. Rio, Y. Maurel, Y. Bugni, O. Barais, Benefits of Energy Management Systems on local energy efficiency, an agricultural case study, *in: SmartGridComm 2019 - IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids*, IEEE, p. 1–7, Beijing, China, October 2019, [hal:hal-02315327].

[171] M. Rodriguez-Cancio, B. Combemale, B. Baudry, Automatic Microbenchmark Generation to Prevent Dead Code Elimination and Constant Folding, *in: 31st IEEE/ACM International Conference on Automated Software Engineering (ASE 2016), ASE 2016:Proceedings of the 31st IEEE/ACM International Conference on Automated Software Engineering*, Singapore, Singapore, September 2016, [hal:hal-01343818].

[172] M. Rodriguez-Cancio, B. Combemale, B. Baudry, Approximate Loop Unrolling, *in: CF 2019 - ACM International Conference on Computing Frontiers*, ACM, p. 94–105, Alghero, Sardinia, Italy, April 2019, [doi:10.1145/3310273.3323841], [hal:hal-02407868].

[173] J. Sallou, A. Gauvain, J. Bourcier, B. Combemale, J.-R. De Dreuzy, Loop Aggregation for Approximate Scientific Computing, *in: International Conference on Computational Science, Lecture Notes in Computer Science book series*, Springer, p. 141–155, Amsterdam, Netherlands, June 2020, [doi:10.1007/978-3-030-50417-5_11], [hal:hal-02545875].

[174] C. Soto-Valero, A. Benelallam, N. Harrand, O. Barais, B. Baudry, The Emergence of Software Diversity in Maven Central, *in: MSR 2019 - 16th International Conference on Mining Software Repositories*, ACM, p. 333–343, Montreal, Canada, May 2019, [doi:10.1109/MSR.2019.00059], [hal:hal-02080248].

[175] C. Soto-Valero, J. Bourcier, B. Baudry, Detection and Analysis of Behavioral T-patterns in Debugging Activities, *in: MSR 2018 - Mining Software Repositories*, p. 1–4, Gothenburg, Sweden, May 2018, [doi:10.1145/3196398.3196452], [hal:hal-01763369].

[176] K. Suri, J. Cadavid, M. Alférez, S. Dhouib, S. Tucci-Piergiovanni, Modeling business motivation and underlying processes for RAMI 4.0-aligned cyber-physical production systems, *in: 2017 22nd IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, IEEE, Limassol, France, September 2017, [doi:10.1109/ETFA.2017.8247702], [hal:hal-01795144].

[177] P. Temple, M. Acher, J.-M. Jézéquel, Multimorphic Testing, *in: ICSE '18 - ACM/IEEE 40th International Conference on Software Engineering*, p. 1–2, Gothenburg, Sweden, May 2018, [doi:10.1145/3183440.3195043], [hal:hal-01730163].

[178] P. Temple, M. Acher, G. Perrouin, B. Biggio, J.-M. Jézéquel, F. Roli, Towards Quality Assurance of Software Product Lines with Adversarial Configurations, *in: SPLC 2019 - 23rd International Systems and Software Product Line Conference*, ACM, p. 277–288, Paris, France, September 2019, [doi:10.1145/3336294.3336309], [hal:hal-02287616].

[179] P. Temple, J. A. Galindo Duarte, M. Acher, J.-M. Jézéquel, Using Machine Learning to Infer Constraints for Product Lines, *in: Software Product Line Conference (SPLC)*, Beijing, China, September 2016, [doi:10.1145/2934466.2934472], [hal:hal-01323446].

[180] L. Thomas Van Binsbergen, M. V. Merino, P. Jeanjean, T. Van Der Storm, B. Combemale, O. Barais, A principled approach to REPL interpreters, *in: SPLASH 2020 - ACM SIGPLAN conference on Systems, Programming, Languages, and Applications: Software for Humanity*, ACM, p. 1–17, Chicago / Virtual, United States, November 2020, [doi:10.1145/3426428.3426917], [hal:hal-02968938].

[181] M. Tricoire, O. Barais, M. Leduc, J. Bourcier, F. Fouquet, G. Nain, L. Mouline, G. Sunyé, B. Morin, KevoreeJS: Enabling Dynamic Software Reconfigurations in the Browser, *in: WICSA and CompArch 2016*, ACM Sigsoft, Venice, Italy, April 2016, [doi:10.1109/CBSE.2016.20], [hal:hal-01354997].

[182] A. Vastel, P. Laperdrix, W. Rudametkin, R. Rouvoy, FP-STALKER: Tracking Browser Fingerprint Evolutions, *in: IEEE S&P 2018 - 39th IEEE Symposium on Security and Privacy*, B. Parno, C. Kruegel (editors), *Proceedings of the 39th IEEE Symposium on Security and Privacy (S&P)*, IEEE, p. 728–741, San Francisco, United States, May 2018, [doi:10.1109/SP.2018.00008], [hal:hal-01652021].

[183] O. L. Vera-Pérez, M. Monperrus, B. Baudry, Descartes: a PITest engine to detect pseudo-tested methods - Tool Demonstration, *in: ASE 2018 - 33rd ACM/IEEE International Conference on Automated Software Engineering, Tool demonstration track*, ACM Press, p. 908–911, Montpellier, France, September 2018, [doi:10.1145/3238147.3240474], [hal:hal-01870976].

[184] M. Vinueza, J. L. Rodas, J. A. Galindo Duarte, D. Benavides, El uso de modelos de características con atributos para pruebas en sistemas de alta variabilidad: primeros pasos, *in: CEDI 2016*, Salamanca, Spain, September 2016, [hal:hal-01342354].

[185] A. Wortmann, B. Combemale, O. Barais, A Systematic Mapping Study on Modeling for Industry 4.0, *in: MODELS '17: Proceedings of the ACM/IEEE 20th International Conference on Model Driven Engineering Languages and Systems*, IEEE Press, Austin, TX, USA, United States, September 2017, [hal:hal-03284857].

National Conferences

[186] A. Blouin, J.-M. Jézéquel, Programmez vos IHM avec Interacto: une démonstration, *in: GDR GPL 2021 - Journées du Groupement de Recherche du Génie de la Programmation et du Logiciel*, p. 1–2, Virtuelle, France, June 2021, [hal:hal-03259896].

[187] J. L. Rodas, J. Olivares, J. A. Galindo Duarte, D. Benavides, Hacia el uso de sistemas de recomendación en sistemas de alta variabilidad, *in: CEDI 2016*, Salamanca, Spain, September 2016, [hal:hal-01342353].

Scientific Books or Book Chapters)

[188] B. Baudry, Y.-D. Bromberg, D. Frey, A. Gómez-Boix, P. Laperdrix, F. Taïani, Profilage de navigateurs : état de l'art et contre-mesures, *in: Le profilage en ligne : entre libéralisme et régulation*, S. Turgis, A. Bensamoun, and M. Boizard (editors), Mare et Martin, October 2020, [hal:hal-03043187].

[189] J. F. F. Bosco, M. Acher, O. Barais, Software Unbundling: Challenges and Perspectives, *in: Transactions on Modularity and Composition I*, S. Chiba, M. Südholt, P. Eugster, L. Ziarek, and G. Leavens (editors), Springer, May 2016, [hal:hal-01427560].

[190] B. Combemale, R. France, J.-M. Jézéquel, B. Rumpe, J. R. Steel, D. Vojtisek, Engineering Modeling Languages : Turning Domain Knowledge into Tools, Chapman and Hall/CRC, November 2016, [hal:hal-01355374].

[191] B. Combemale, M. Wimmer, Towards a Model-Based DevOps for Cyber-Physical Systems, *in: Software Engineering Aspects of Continuous Development*, Springer-Verlag, 2019, p. 1–11, [hal:hal-02407886].

[192] B. Coppens, O. Zendra, Privacy: whether you're aware of it or not, it does matter!, *in: HiPEAC Vision 2021*, HiPEAC (editor), January 2021, [doi:10.5281/zenodo.4719402], [hal:hal-03362809].

[193] T. Degueule, B. Combemale, J.-M. Jézéquel, On Language Interfaces, *in: PAUSE: Present And Ulterior Software Engineering*, B. Meyer and M. Mazzara (editors), Springer, February 2017, [hal:hal-01424909].

[194] M. Duranton, K. De Bosschere, B. Coppens, C. Gamrat, t. hoberg, H. Munk, C. Roderick, T. Vardanega, O. Zendra, HiPEAC Vision 2021, January 2021, [hal:hal-03359519].

[195] P. Laperdrix, G. Avoine, B. Baudry, N. Nikiforakis, Morellian Analysis for Browsers: Making Web Authentication Stronger with Canvas Fingerprinting, *in: Detection of Intrusions and Malware, and Vulnerability Assessment - 16th International Conference, DIMVA 2019, Gothenburg, Sweden, June 19-20, 2019, Proceedings*, June 2019, p. 43–66, [doi:10.1007/978-3-030-22038-9_3], [hal:hal-02881632].

[196] R. E. Lopez-Herrejon, T. Ziadi, J. Martinez, M. Acher, Fourth International Workshop on Reverse Variability Engineering (REVE 2016), associated with SPLC, September 2016, [hal:hal-01628959].

[197] J. Pelay, F. Guillemin, O. Barais, Verifying the configuration of Virtualized Network Functions in Software Defined Networks, b<>com, 2017, Accepted version of the paper but with two formulas which had to be deleted in the final version for size reasons., [hal:hal-01657866].

[198] D. Weyns, N. Bencomo, R. Calinescu, J. Cámara, C. Ghezzi, V. Grassi, L. Grunske, P. Inverardi, J.-M. Jézéquel, S. Malek, R. Mirandola, M. Mori, G. Tamburrelli, Perpetual Assurances for Self-Adaptive Systems, *in: Software Engineering for Self-Adaptive Systems III. Assurances*, 2017, https://arxiv.org/abs/1903.04771, [hal:hal-02054220].

[199] O. Zendra, B. Coppens, Cybersecurity must come to IT systems now, *in: HiPEAC Vision 2021*, HiPEAC (editor), January 2021, [hal:hal-03362808].

[200] O. Zendra, K. De Bosschere, Taming the IT systems complexity hydra, *in: HiPEAC Vision 2021*, HiPEAC (editor), January 2021, [doi:10.5281/zenodo.4719574], [hal:hal-03362810].

Edition of Books or Proceedings

[201] Report on the 1 st International Workshop on Debugging in Model-Driven Engineering (MDEbug'17), Austin, United States, December 2017, 1–6p., [hal:hal-01665572].

Research Reports

[202] M. Acher, F. Esnault, Large-scale Analysis of Chess Games with Chess Engines: A Preliminary Report, *Technical Report number RT-0479*, Inria Rennes Bretagne Atlantique, April 2016, https://arxiv.org/abs/1607.04186, [hal:hal-01307091].

[203] M. Acher, H. Martin, J. Alves Pereira, A. Blouin, D. Eddine Khelladi, J.-M. Jézéquel, Learning From Thousands of Build Failures of Linux Kernel Configurations, *Technical report*, Inria ; IRISA, June 2019, [hal:hal-02147012].

[204] M. Acher, H. Martin, J. A. Pereira, A. Blouin, J.-M. Jézéquel, D. E. Khelladi, L. Lesoil, O. Barais, Learning Very Large Configuration Spaces: What Matters for Linux Kernel Sizes, *Research report*, Inria Rennes - Bretagne Atlantique, October 2019, [hal:hal-02314830].

[205] J. Alves Pereira, H. Martin, M. Acher, J.-M. Jézéquel, G. Botterweck, A. Ventresque, Learning Software Configuration Spaces: A Systematic Literature Review, *Research Report number 1-44*, Univ Rennes, Inria, CNRS, IRISA, June 2019, https://arxiv.org/abs/1906.03018, [doi:10.1145/nnnnnnn.nnnnnnn], [hal:hal-02148791].

[206] G. Le Guernic, J. A. Galindo Duarte, Experience Report on the Formal Specification of a Packet Filtering Language Using the K Framework, *Research Report number RR-8967*, Inria Rennes Bretagne Atlantique, October 2016, [hal:hal-01385541].

[207] G. Le Guernic, Experience Report on the Development of a Specialized Multi-view Multi-stakeholder Model-Based Engineering Framework (extended version), *Research Report number RR-9283*, Inria Rennes - Bretagne Atlantique ; IRISA, December 2019, [hal:hal-02398051].

[208] G. Lyan, D. Gross-Amblard, J.-M. Jézéquel, S. Malinowski, Impact of Data Cleansing for Urban Bus Commercial Speed Prediction, *Research report*, Université de Rennes ; IRISA ; Keolis Rennes, May 2021, [hal:hal-03220449].

[209] P. Temple, M. Acher, J.-M. Jézéquel, L. Noel-Baron, J. A. Galindo, Learning-Based Performance Specialization of Configurable Systems, *Research report*, IRISA, Inria Rennes ; University of Rennes 1, February 2017, [hal:hal-01467299].

[210] A. Wortmann, B. Combemale, O. Barais, A Systematic Mapping Study on Modeling for Industry 4.0, *Research Report number RR-9062*, INRIA Rennes - Bretagne Atlantique and University of Rennes 1, France, April 2017, [hal:hal-01514421].

Miscellaneous

[211] T. Degueule, Interoperability and Composition of DSLs with Melange, working paper or preprint, June 2016, [hal:hal-01336940].

[212] M. Monperrus, B. Danglot, O. L. Vera-Perez, Z. Yu, B. Baudry, The Emerging Field of Test Amplification: A Survey, https://arxiv.org/abs/1705.10692 - working paper or preprint, November 2017, [hal:hal-01634288].

[213] A. Rio, Y. Maurel, O. Barais, Y. Bugni, Demand Side Management A model driven approach to promote energy self-consumption, Colloque de l'institut Mines Télécom, June 2018, Poster, [hal:hal-02317918].

[214] M. Rodriguez-Cancio, B. Combemale, B. Baudry, Approximate Loop Unrolling, working paper or preprint, November 2016, [hal:hal-01401828].

Additional references

[AB11] Andrea Arcuri and Lionel C. Briand. A practical guide for using statistical tests to assess randomized algorithms in software engineering. In *ICSE*, pages 1–10, 2011.

[Avi85] Algirdas Avizienis. The n-version approach to fault-tolerant software. *Software Engineering, IEEE Transactions on*, (12):1491–1501, 1985.

[BAC+99] Lionel Claude Briand, Erik Arisholm, Steve Counsell, Frank Houdek, and Pascale Thévenod-Fosse. Empirical studies of object-oriented artifacts, methods, and processes: state of the art and future directions. *Empirical Software Engineering*, 4(4):387–404, 1999.

[BAFS05] Elena Gabriela Barrantes, David H Ackley, Stephanie Forrest, and Darko Stefanović. Randomized instruction set emulation. *ACM Transactions on Information and System Security (TISSEC)*, 8(1):3–40, 2005.

[BB01] F. Bachmann and L. Bass. Managing variability in software architectures. *SIGSOFT Softw. Eng. Notes*, 26:126–132, May 2001.

[BC04] Elisa Baniassad and Siobhàn Clarke. Theme: an approach for aspect-oriented analysis and design. In *26th International Conference on Software Engineering (ICSE)*, pages 158–167, 2004.

[Ben09] Nelly 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.

[BFG+02] Jan Bosch, Gert Florijn, Danny Greefhorst, Juha Kuusela, J. Henk Obbink, and Klaus Pohl. Variability issues in software product lines. In *PFE '01: Revised Papers from the 4th International Workshop on Software Product-Family Engineering*, pages 13–21, London, UK, 2002. Springer-Verlag.

[BHLM94] Joseph T Buck, Soonhoi Ha, Edward A Lee, and David G Messerschmitt. Ptolemy: A framework for simulating and prototyping heterogeneous systems. *Int. Journal of Computer Simulation*, 1994.

[BHP06] Tomas Bures, Petr Hnetynka, and Frantisek Plasil. Sofa 2.0: Balancing advanced features in a hierarchical component model. In *Software Engineering Research, Management and Applications, 2006. Fourth International Conference on*, pages 40–48. IEEE, 2006.

[BJP10] Antoine Beugnard, Jean-Marc Jézéquel, and Noël Plouzeau. Contract aware components, 10 years after. In *WCSI*, pages 1–11, 2010.

[BKR09] Steffen Becker, Heiko Koziolek, and Ralf Reussner. The Palladio component model for model-driven performance prediction. *Journal of Systems and Software*, 82(1):3–22, January 2009.

[BLHM02] Don Batory, Roberto E Lopez-Herrejon, and Jean-Philippe Martin. Generating product-lines of product-families. In *ASE '02: Automated software engineering*, pages 81–92. IEEE, 2002.

[Bos00] Jan 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.

[BWH+03] Felice Balarin, Yosinori Watanabe, Harry Hsieh, Luciano Lavagno, Claudio Passerone, and Alberto Sangiovanni-Vincentelli. Metropolis: An integrated electronic system design environment. *Computer*, 36(4):45–52, 2003.

[CHW98] James Coplien, Daniel Hoffman, and David Weiss. Commonality and variability in software engineering. *IEEE Software*, 15(6):37–45, 1998.

[CLG+09] Betty H. C. Cheng, Rogério Lemos, Holger Giese, Paola Inverardi, Jeff Magee, Jesper Andersson, Basil Becker, Nelly Bencomo, Yuriy Brun, Bojan Cukic, Giovanna Marzo Serugendo, Schahram Dustdar, Anthony Finkelstein, Cristina Gacek, Kurt Geihs, Vincenzo Grassi, Gabor Karsai, Holger M Kienle, Jeff Kramer, Marin Litoiu, Sam Malek, Raffaela Mirandola, Hausi A Müller, Sooyong Park, Mary Shaw, Matthias Tichy, Massimo Tivoli, Danny Weyns, and Jon Whittle. *Software Engineering for Self-Adaptive Systems: A Research Roadmap* , volume 5525 of *Betty H. C. Cheng, Rogério de Lemos, Holger Giese, Paola Inverardi, and Jeff Magee.* Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.

[CSVC11] Ivica Crnkovic, Séverine Sentilles, Aneta Vulgarakis, and Michel RV Chaudron. A classification framework for software component models. *Software Engineering, IEEE Transactions on*, 37(5):593–615, 2011.

[DCB+15] Thomas Degueule, Benoit Combemale, Arnaud Blouin, Olivier Barais, and Jean-Marc 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.

[DO91] RA DeMilli and A. Jefferson Offutt. Constraint-based automatic test data generation. *Software Engineering, IEEE Transactions on*, 17(9):900–910, 1991.

[DPAM02] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and TAMT Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *Evolutionary Computation, IEEE Transactions on*, 6(2):182–197, 2002.

[FFH13] Sören Frey, Florian Fittkau, and Wilhelm Hasselbring. Search-based genetic optimization for deployment and reconfiguration of software in the cloud. In *Proceedings of the 2013 International Conference on Software Engineering*, pages 512–521. IEEE Press, 2013.

[FR07] Robert B. France and Bernhard Rumpe. Model-driven development of complex software: A research roadmap. In Lionel C. Briand and Alexander L. Wolf, editors, *Proceedings of the Future of Software Engineering Symposium (FOSE '07)*, pages 37–54. IEEE, 2007.

[FSA97] Stephanie Forrest, Anil Somayaji, and David H Ackley. Building diverse computer systems. In *Operating Systems, 1997., The Sixth Workshop on Hot Topics in*, pages 67–72. IEEE, 1997.

[HB08] Cécile Hardebolle and Frédéric Boulanger. Modhel'x: A component-oriented approach to multi-formalism modeling. In *Models in Software Engineering*, pages 247–258. Springer, 2008.

[HBAA10] Hadi Hemmati, Lionel C. Briand, Andrea Arcuri, and Shaukat Ali. An enhanced test case selection approach for model-based testing: an industrial case study. In *SIGSOFT FSE*, pages 267–276, 2010.

[HJ01] Mark Harman and Bryan F Jones. Search-based software engineering. *Information and Software Technology*, 43(14):833–839, 2001.

[HP03] Günter Halmans and Klaus Pohl. Communicating the variability of a software-product family to customers. *Software and System Modeling*, 2(1):15–36, 2003.

[HWRK11] John Hutchinson, Jon Whittle, Mark Rouncefield, and Steinar Kristoffersen. Empirical assessment of MDE in industry. In Richard N. Taylor, Harald Gall, and Nenad Medvidovic, editors, *Proceedings of the 33rd International Conference on Software Engineering (ICSE '11)*, pages 471–480. ACM, 2011.

[Jéz08] Jean-Marc Jézéquel. Model driven design and aspect weaving. *Journal of Software and Systems Modeling (SoSyM)*, 7(2):209–218, may 2008.

[KCH+90] K. C. Kang, S. G. Cohen, J. A. Hess, W. E. Novak, and A. S. Peterson. Feature-oriented domain analysis (foda) feasibility study. Technical report, Carnegie-Mellon University Software Engineering Institute, November 1990.

[KM07] Jeff Kramer and Jeff Magee. Self-Managed Systems: an Architectural Challenge. In *Future of Software Engineering*, pages 259–268. IEEE, 2007.

[LEW05] Kung-Kiu Lau, Perla Velasco Elizondo, and Zheng Wang. Exogenous connectors for software components. In *Component-Based Software Engineering*, pages 90–106. Springer, 2005.

[MBJ+09] Brice Morin, Olivier Barais, Jean-Marc Jézéquel, Franck Fleurey, and Arnor Solberg. Models at runtime to support dynamic adaptation. *IEEE Computer*, pages 46–53, October 2009.

[McM04] Phil McMinn. Search-based software test data generation: a survey. *Software Testing, Verification and Reliability*, 14(2):105–156, 2004.

[Mem07] Atif M Memon. An event-flow model of gui-based applications for testing. *Software Testing, Verification and Reliability*, 17(3):137–157, 2007.

[MFJ05] P.-A. Muller, F. Fleurey, and Jean-Marc Jézéquel. Weaving executability into object-oriented meta-languages. In *Proc. of MODELS/UML'2005*, LNCS, Jamaica, 2005. Springer.

[MHM98] Jacques Meekel, Thomas B. Horton, and Charlie Mellone. Architecting for domain variability. In *ESPRIT ARES Workshop*, pages 205–213, 1998.

[MMR+10] Rémi Mélisson, Philippe Merle, Daniel Romero, Romain Rouvoy, and Lionel Seinturier. Reconfigurable run-time support for distributed service component architectures. In *the IEEE/ACM international conference*, page 171, New York, New York, USA, 2010. ACM Press.

[NLTJ06] Clémentine Nebut, Yves Le Traon, and Jean-Marc Jézéquel. *System Testing of Product Families: from Requirements to Test Cases*, pages 447–478. Springer Verlag, 2006.

[Nor99] Linda M. Northrop. A framework for software product line practice. In *Proceedings of the Workshop on Object-Oriented Technology*, pages 365–366, London, UK, 1999. Springer-Verlag.

[Nor02] Linda M. Northrop. SEI's software product line tenets. *IEEE Softw.*, 19(4):32–40, 2002.

[NPLJ03] Clémentine Nebut, Simon Pickin, Yves Le Traon, and Jean-Marc 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.

[OGO06] Iulian Ober, Susanne Graf, and Ileana Ober. Validating timed uml models by simulation and verification. *International Journal on Software Tools for Technology Transfer*, 8(2):128–145, 2006.

[Par76] D. L. Parnas. On the design and development of program families. *IEEE Trans. Softw. Eng.*, 2(1):1–9, 1976.

[PBv05] Klaus Pohl, Günter Böckle, and Frank J. van der Linden. *Software Product Line Engineering: Foundations, Principles and Techniques.* Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2005.

[PJJ+07] Simon Pickin, Claude Jard, Thierry Jéron, Jean-Marc Jézéquel, and Yves Le Traon. Test synthesis from UML models of distributed software. *IEEE Transactions on Software Engineering*, 33(4):252–268, April 2007.

[Ran75] Brian Randell. System structure for software fault tolerance. *Software Engineering, IEEE Transactions on*, (2):220–232, 1975.

[RH09] Per Runeson and Martin Höst. Guidelines for conducting and reporting case study research in software engineering. *Empirical Software Engineering*, 14(2):131–164, 2009.

[Rin12] Martin Rinard. Obtaining and reasoning about good enough software. In *Proceedings of Annual Design Automation Conference (DAC)*, pages 930–935, 2012.

[RWLN89] Jeff Rothenberg, Lawrence E. Widman, Kenneth A. Loparo, and Norman R. Nielsen. The nature of modeling. In *in Artificial Intelligence, Simulation and Modeling*, pages 75–92. John Wiley & Sons, 1989.

[SBMP08] Dave Steinberg, Frank Budinsky, Ed Merks, and Marcelo Paternostro. *EMF: eclipse modeling framework.* Pearson Education, 2008.

[Sch06] D.C. Schmidt. Guest editor's introduction: Model-driven engineering. *IEEE Computer*, 39(2):25–31, 2006.

[SDMHR11] Stelios Sidiroglou-Douskos, Sasa Misailovic, Henry Hoffmann, and Martin Rinard. Managing performance vs. accuracy trade-offs with loop perforation. In *Proc. of the Symp. on Foundations of software engineering*, ESEC/FSE '11, pages 124–134, New York, NY, USA, 2011. ACM.

[SGM02] Clemens Szyperski, Dominik Gruntz, and Stephan Murer. *Component software: beyond object-oriented programming.* Addison-Wesley, 2002.

[SJ07] Jim Steel and Jean-Marc Jézéquel. On model typing. *Journal of Software and Systems Modeling (SoSyM)*, 6(4):401–414, December 2007.

[SSS08] Forrest Shull, Janice Singer, and Dag IK Sjberg. *Guide to advanced empirical software engineering*. Springer, 2008.

[TH03] Jean-Christophe Trigaux and Patrick Heymans. Modelling variability requirements in software product lines: a comparative survey. Technical report, FUNDP Namur, 2003.

[UL10] Mark Utting and Bruno Legeard. *Practical model-based testing: a tools approach*. Morgan Kaufmann, 2010.

[VWMA11] Pieter Vromant, Danny Weyns, Sam Malek, and Jesper Andersson. On interacting control loops in self-adaptive systems. In *SEAMS 2011*, pages 202–207. ACM, 2011.

[YCP06] Cemal Yilmaz, Myra B Cohen, and Adam A Porter. Covering arrays for efficient fault characterization in complex configuration spaces. *Software Engineering, IEEE Transactions on*, 32(1):20–34, 2006.

[ZJ06] Tewfik Ziadi and Jean-Marc Jézéquel. *Product Line Engineering with the UML: Deriving Products*, pages 557–586. Springer Verlag, 2006.

[ZMKR12] Zeyuan Allen Zhu, Sasa Misailovic, Jonathan A. Kelner, and Martin Rinard. Randomized accuracy-aware program transformations for efficient approximate computations. In *Proc. of the Symp. on Principles of Programming Languages (POPL)*, pages 441–454, 2012.

A DiverSE assessment *by DiverSE*: SWOT analysis

In this section, we share and discuss our vision regarding our **Strengths**, **Weaknesses**, **Opportunities** and **Threats**.

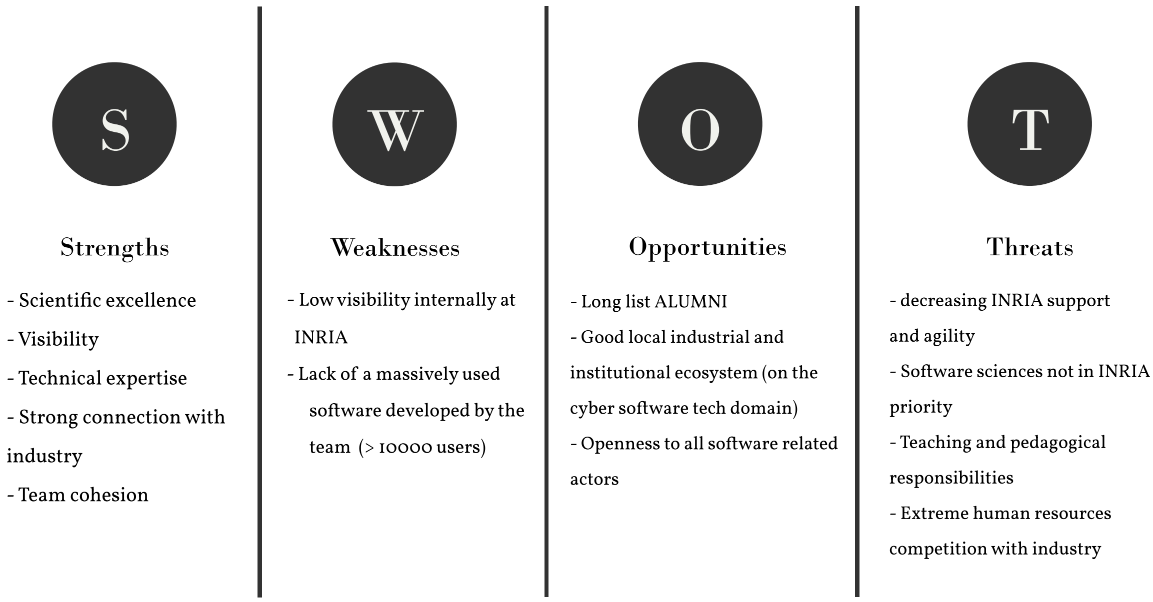

Figure 7: DiverSE SWOT Analysis

A.1 Strengths

Our two major strengths that we wish to highlight are our scientific excellence and our academic visibility. We have been a dynamic and internationally recognized research group in the field of software engineering, for over 20 years. We are very active in the international community, and we regularly publish at the highest level venues while regularly training PhDs able of being recruited as research fellows or able to lead research and technical activities in industry.

Another strong asset of the team is its technical expertise. With this technical expertise we are able to understand modern software engineering challenges. It also allows us to be relevant with our industrial partners.

Finally, team cohesion is also one of our core strengths. We work as an efficient team, sharing our scientific challenges, our results, our teaching material, our PhD students, our financial resources, etc. We believe in the strength of a collaborative research team that also promotes individual recognition, e.g. CNRS silver medal, IUF (Institut Universitaire de France), career evolution, etc.

A.2 Weaknesses

Our main weakness is our lack of internal visibility at INRIA. We strongly focus on international cooperation and competition within the academic community, while we rarely participate in INRIA's initiatives. Olivier Zendra arrival (an INRIA Evaluation Committee member) should help us improve this situation.

Our second weakness is the lack of a software developed by the team members that would both highlight our expertise and impact a very large community of users. Even though some libraries developed within the team are still massively downloaded (they are used by software that are themselves heavily used), we still have the ambition to find the energy to either develop a general purpose software with a large audience using our tools or to succeed in having one of our research prototypes reaching a highly advanced TRL and being massively used by developers.

A.3 Opportunities

Our main opportunity is our long list of alumni. Former members of Triskell[16] or DiverSE share fond memories of their time in the group. Whether in Luxembourg, Norway, Spain, Sweden, USA or within industrial players such as Mathworks, Amazon, Orange, IBM, CEA, XWiki, MediaKind, OKWind, Obéo, Datathings, EXFO, FAMOCO, Criteo, energensys, etc. these former team members are gateways to effective collaborations or/and international recruitment.

Our second opportunity is the excellent ecosystem in the field of computer science, in particular in the field of software engineering and software security in western France. This ecosystem is a positive source to attract future talents from, but also enables setting up fruitful local partnerships.

Our final opportunity is our large expertise all along the software lifecycle and the corresponding software industry. Our understanding of the entire ecosystem, from requirements to deployment, including testing, design and evolution issues, allows us to propose systemic approaches that are best integrated into modern software development processes.

A.4 Threats

From our point of view, the main threat is the loss of culture at INRIA in core software sciences and engineering. The idea that activities in the field of software are "just engineering" and that software should be "at the service of a scientific domain" is from our point of view the main threat to the DiverSE EPI. We thus fully agree with the text [éléments de réflexion sur la stratégie scientifique d'un institut de recherche en informatique, automatique et mathématiques appliquées. Document submitted to the Inria Evaluation Commission on 30/09/2021, prepared by the working group composed of Céline Grandmont (coordinator), Valérie Issarny (coordinator), Emmanuel Thomé (coordinator), Anne Canteaut, Benjamin Guedj, Jean-Marc Lasgouttes, Nathalie Mitton, Guillaume Pallez, Vivien Quéma, Xavier Rival, Manuel Serrano, Jean-Bernard Stefani.

The second main threat to our team is the significant decrease in resources at the University in the face of a constant increase in the number of students in software sciences. As an example, the electronics and computing department of the University of Rennes went from 1000 to 1600 students in the last ten years, all without any additional resource. This results in a heavy administrative burden for the associate professors and professors. The majority of DiverSE permanent staff is made of professors or associate professors, so this overload has a huge impact on our team. For the French readers, we would like to highlight two recent press articles which point out the phenomenon[17][18].

The third important threat is the strong competition with international academic institute or industry for the recruitment of doctoral students, post-doctoral fellows or researchers. At the academic level, Canada, Luxembourg, Sweden and Switzerland have implemented aggressive recruitment policies to attract researchers. Very clearly, French salaries, research fundings and other critical research means are strongly unattractive to candidates. Another facet of this threat is that the salaries proposed by companies to our doctoral students at the end of their PhDs are close to those of a research director at the end of his or her career, making companies in the digital sector extremely strong competitors in the recruitment field. (We acknowledge that this phenomenon is not limited to software engineering domain.)

Our last threat is the decreasing agility of INRIA in all administrative procedures. For a long time protected from the administrative burden inherent to any large structure, we have

[16] Former name of the team
[17] https://shorturl.at/dnENW
[18] https://shorturl.at/ozFP7

observed a strong degradation of the situation at INRIA for several years. Optim (an internal reorganization) has moved the services further away from the teams, and in some aspects, the University, once considered the epitome of administrative burden, is becoming more agile than INRIA.

We wish to share this SWOT with the internal and external reviewers. We would like to recall that during the previous INRIA evaluation, we pointed out the threat of the possible departure of our brilliant INRIA researchers due to the absence of an internal human resource policy at INRIA. Two years later, the leader of the DiverSE team, (CR INRIA, holder of two European projects), left France for Sweden. This threat remains present for the group, several talented associate professors hold their "*habilitation à diriger des recherches*". If INRIA or the University of Rennes are not able to offer them visibility on their career development, there is no doubt that our foreign competitors will take advantage of this weakness.

B DiverSE scientific background

B.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) [Sch06]. 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 [FR07]. 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 [BC04]. 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 in the development of a sound *Software Language Engineering* [19], including a unified typing theory that integrate models as first class entities [SJ07].

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. [HWRK11] indicates 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 development and study of appropriate mechanisms for continuous design and runtime 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 runtime behavior is extremely promising [MBJ$^+$09].

B.2 Variability modeling

While the basic vision underlying *Software Product Lines* (SPL) can probably be traced back to David Parnas' seminal article [Par76] 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 [Nor99]. 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*

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

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 [Nor02]. Bosch provides a different definition [Bos00]: *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 [CHW98]. 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 [HP03]. 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 [BFG+02]. Variability management is thus growingly seen as the cornerstone of SPL development, covering the entire development life cycle, from requirements elicitation [TH03] to product derivation [ZJ06] to product testing [NPLJ03, NLTJ06].

Halmans *et al.* [HP03] 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.* [PBv05]: 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.* [MHM98] (feature, hardware platform, performances and attributes variability) or Bass *et al.* [BB01] who discusses 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* [KCH+90]. 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 [BFG+02]. 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 [BLHM02]. Model Driven Engineering (MDE) makes it possible to automate this weaving process [Jéz08]. 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 [MFJ05]) manipulating these models to produce for instance a detailed design that can ultimately be transformed to code, or to test suites [PJJ+07], or other software artifacts.

B.3 Component-based software development

Component-based software development [SGM02] 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 [CSVC11]: description of requirements on services required from the other components; indirect connections between components thanks to ports and connectors constructs [LEW05]; hierarchical definition of components (assemblies of components can define new component types); connectors supporting various communication semantics [BHP06]; quantitative properties on the services [BJP10].

In recent years component-based architectures have evolved from static designs to dynamic, adaptive designs (e.g., SOFA [BHP06], Palladio [BKR09], Frascati [MMR+10]). 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 a stop/redeploy/start process). Dynamic adaptation processes support *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 [KM07]. 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 runtime, without human intervention, by adapting themselves [CLG+09, VWMA11]. 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 [Ben09], dynamic component based systems must separate architectural design and platform compatibility. This requires support for heterogeneity, since platform evolution 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 [RWLN89]". In practice, component-based (and/or service-based) platforms offer reflection APIs that make it possible to introspect the system (to determine 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 [HJ01] 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 [FFH13]. Multi Objectives Search based techniques [DPAM02] 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.

B.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 [UL10] mainly relies on model analysis, constraint solving [DO91] and search-based reasoning [McM04]. DiverSE leverages in particular the applications of model-based testing in the context of highly-configurable systems and [YCP06] interactive systems [Mem07] as well as recent advances based on diversity for test cases selection [HBAA10].

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 [OGO06], or Xholon [20]. 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 [BHLM94] 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 [BWH+03] provides modeling elements amenable to semantically equivalent mathematical models. Metropolis offers a precise semantics flexible enough to support different models of computation. ModHel'X [HB08] studies the composition of multi-paradigm models relying on different models of computation.

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

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 [Avi85], recovery blocks [Ran75] and code randomization [BAFS05], 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.

B.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 of 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 [SSS08, RH09]. Such methods have been used for example to understand the impact of new software development paradigms [BAC$^+$99]. 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 [AB11].