IN PARTNERSHIP WITH:
Université des sciences et technologies de Lille (Lille 1)

Activity Report 2016

Project-Team RMOD

Analyses and Languages Constructs for Object-Oriented Application Evolution

IN COLLABORATION WITH: Centre de Recherche en Informatique, Signal et Automatique de Lille

RESEARCH CENTER
Lille - Nord Europe

THEME
Distributed programming and Software engineering

Table of contents

Project-Team RMOD

Creation of the Project-Team: 2009 July 01

Keywords:

Computer Science and Digital Science:

2. - Software
2.1. - Programming Languages
2.1.2. - Object-oriented programming
2.1.4. - Aspect-oriented programming
2.1.9. - Dynamic languages
2.1.10. - Domain-specific languages
2.5. - Software engineering
2.5.1. - Software Architecture & Design
2.5.3. - Empirical Software Engineering
2.5.4. - Software Maintenance & Evolution
2.6. - Infrastructure software
2.6.3. - Virtual machines

Other Research Topics and Application Domains:

2. - Health
2.7. - Medical devices
5. - Industry of the future
5.9. - Industrial maintenance
6.5. - Information systems
7. - Transport and logistics

1. Members

Research Scientists

Stéphane Ducasse [Team leader, Inria, Research Scientist, HDR]
Marcus Denker [Inria, Research Scientist]

Faculty Members

Nicolas Anquetil [Univ. Lille I, Faculty Member, HDR]
Damien Cassou [Univ. Lille I, Faculty Member, left for industry in Jul 2016]
Anne Etien [Univ. Lille I, Faculty Member, HDR]
Damien Pollet [Univ. Lille I, Faculty Member]

Engineers

Christophe Demarey [Inria, 70%]
Guillermo Polito [CNRS, 60%]
Olivier Auverlot [Univ. Lille I, 30%]
Pavel Krivanek [Inria, since Mar 2016]
Denis Kudriashov [Inria, since Oct 2015]
Esteban Lorenzano [Inria]
Nicolas Passerini [Inria, since Apr 2016]

PhD Students

Clément Béra [Inria, 2rd year, granted by Region Nord Pas de Calais]
Vincent Blondeau [Worldline, 2rd year, granted by CIFRE]
Gustavo Jansen de Souza Santos [Inria, 3rd year, granted by CNPq (Brazil)]
Brice Govin [Thales, 2rd year, granted by CIFRE]
Thibault Raffaillac [Inria, 2nd year, co-supervision with Stéphane Huot of Mjolnir team Inria]
Pablo Tesone [Inria-Ecole des Mines de Douai, 1st year]
Jason Lecerf [CEA-LIST, 1st year, co-supervision with Thierry Goubier]
Marco Naddeo [University of Torino, 3rd year, co-supervision with Pr. Viviana Bono]

Visiting Scientists

Klérisson Vinicius Ribeiro Da Paixão [University of Uberlandia (Brazil), 10 months internship from Oct 2015 to Jul 2016]
Abdelghani Alidra [Badji Mokhtar University (Algeria), 3 months internship from Nov 2015 to Jan 2016]
Cedrik Beler [ENIT, Apr 2016]
Alexandre Bergel [University of Chile]
Johan Fabry [University of Chile]
Ronie Salgado Faila [University of Chile, PhD Student, 6 months internship from Apr 2016 to Sep 2016]
Peter Uhnak [Czech Technical University in Prague, Master Student, from Apr 2016 to May 2016]

Administrative Assistant

Julie Jonas [Inria]

Others

Lionel Akue [Université de Lomé, Software Engineer, from Oct 2016 to Nov 2016]
Thibault Arloing [Univ. Lille I, Master student, from Apr 2016 until Sep 2016]
Julien Delplanque [Inria, Master Student, from Oct 2016 to Dec 2016]
Yann Dubois [Univ. Lille I, Master student, from Apr 2016 until Sep 2016]
Luc Fabresse [Ecole des Mines de Douai, Faculty Member, from Apr 2016]
Thomas Heniart [Univ. Lille I, Master student, until Jul 2016]
Skip Lentz [Inria, Master student, until Jan 2016]
Marion Noirbent [Univ. Lille I, Master student, from Apr 2016 until Aug 2016]
Maxime Roelandt [Univ. Lille I, Master student, from Apr 2016 until Sep 2016]
Valentin Ryckewaert [Univ. Lille I, Master student, from Apr 2016 until Aug 2016]

2. Overall Objectives

2.1. Introduction

Keywords: Software evolution, Maintenance, Program visualization, Program analyses, Meta modelling, Software metrics, Quality models, Object-oriented programming, Reflective programming, Traits, Dynamically typed languages, Dynamic Software Update, Pharo, Moose.

RMoD's general vision is defined in two objectives: remodularization and modularity constructs. These two views are the two faces of a same coin: maintenance could be eased with better engineering and analysis tools and programming language constructs could let programmers define more modular applications.

2.2. Reengineering and remodularization

While applications must evolve to meet new requirements, few approaches analyze the implications of their original structure (modules, packages, classes) and their transformation to support their evolution. Our research focuses on the *remodularization* of object-oriented applications. Automated approaches including clustering algorithms are not satisfactory because they often ignore user inputs. Our vision is that we need better approaches to support the transformation of existing software. The reengineering challenge tackled by RMoD is formulated as follows:

How to help remodularize existing software applications?

We are developing analyses and algorithms to remodularize object-oriented applications. This is why we started studying and building tools to support the *understanding of applications* at the level of packages and modules. This allows us to understand the results of the *analyses* that we are building.

We seek to create tools to help developers perform large refactoring. How can they keep track of changes in various locations in a system while ensuring *integrity of current and new code* by *uniformally applying new design choices*.

2.3. Constructs for modular and isolating programming languages

Dynamically-typed programming languages such as JavaScript are getting new attention as illustrated by the large investment of Google in the development of the Chrome V8 JavaScript engine and the development of a new dynamic language DART. This new trend is correlated to the increased adoption of dynamic programming languages for web-application development, as illustrated by Ruby on Rails, PHP and JavaScript. With web applications, users expect applications to be always available and getting updated on the fly. This continuous evolution of application is a real challenge [54]. Hot software evolution often requires *reflective* behavior and features. For instance in CLOS and Smalltalk each class modification automatically migrates existing instances on the fly.

At the same time, there is a need for *software isolation i.e.,*, applications should reliably run co-located with other applications in the same virtual machine with neither confidential information leaks nor vulnerabilities. Indeed, often for economical reasons, web servers run multiple applications on the same virtual machine. Users need confined applications. It is important that (1) an application does not access information of other applications running on the same virtual machine and (2) an application authorized to manipulate data cannot pass such authorization or information to other parts of the application that should not get access to it.

Static analysis tools have always been confronted to reflection [51]. Without a full treatment of reflection, static analysis tools are both incomplete and unsound. Incomplete because some parts of the program may not be included in the application call graph, and unsound because the static analysis does not take into account reflective features [60]. In reflective languages such as F-Script, Ruby, Python, Lua, JavaScript, Smalltalk and Java (to a certain extent), it is possible to nearly change any aspect of an application: change objects, change classes dynamically, migrate instances, and even load untrusted code.

Reflection and isolation concerns are a priori antagonistic, pulling language design in two opposite directions. Isolation, on the one hand, pulls towards more static elements and types (*e.g.,*, ownership types). Reflection, on the other hand, pulls towards fully dynamic behavior. This tension is what makes this a real challenge: As experts in reflective programming, dynamic languages and modular systems, we believe that by working on this important tension we can make a breakthrough and propose innovative solutions in resolving or mitigating this tension. With this endeavor, we believe that we are working on a key challenge that can have an impact on future programming languages. The language construct challenge tackled by RMoD is formulated as follows:

What are the language modularity constructs to support isolation?

In parallel we are continuing our research effort on traits [1] by assessing trait scalability and reuse on a large case study and developing a pure trait-based language. In addition, we dedicate efforts to remodularizing a meta-level architecture in the context of the design of an isolating dynamic language. Indeed at the extreme, modules and structural control of reflective features are the first steps towards flexible, dynamic, yet isolating, languages. As a result, we expect to demonstrate that having adequate composable units and scoping units will help the evolution and recomposition of an application.

[1] Traits are groups of methods that can be composed orthogonally to simple inheritance. Contrary to mixin, the class has the control of the composition and conflict management.

3. Research Program

3.1. Software Reengineering

Strong coupling among the parts of an application severely hampers its evolution. Therefore, it is crucial to answer the following questions: How to support the substitution of certain parts while limiting the impact on others? How to identify reusable parts? How to modularize an object-oriented application?

Having good classes does not imply a good application layering, absence of cycles between packages and reuse of well-identified parts. Which notion of cohesion makes sense in presence of late-binding and programming frameworks? Indeed, frameworks define a context that can be extended by subclassing or composition: in this case, packages can have a low cohesion without being a problem for evolution. How to obtain algorithms that can be used on real cases? Which criteria should be selected for a given remodularization?

To help us answer these questions, we work on enriching Moose, our reengineering environment, with a new set of analyses [45], [44]. We decompose our approach in three main and potentially overlapping steps:

1. Tools for understanding applications,
2. Remodularization analyses,
3. Software Quality.

3.1.1. Tools for understanding applications

Context and Problems. We are studying the problems raised by the understanding of applications at a larger level of granularity such as packages or modules. We want to develop a set of conceptual tools to support this understanding.

Some approaches based on Formal Concept Analysis (FCA) [75] show that such an analysis can be used to identify modules. However the presented examples are too small and not representative of real code.

Research Agenda.

FCA provides an important approach in software reengineering for software understanding, design anomalies detection and correction, but it suffers from two problems: (i) it produces lattices that must be interpreted by the user according to his/her understanding of the technique and different elements of the graph; and, (ii) the lattice can rapidly become so big that one is overwhelmed by the mass of information and possibilities [34]. We look for solutions to help people putting FCA to real use.

3.1.2. Remodularization analyses

Context and Problems. It is a well-known practice to layer applications with bottom layers being more stable than top layers [61]. Until now, few works have attempted to identify layers in practice: Mudpie [77] is a first cut at identifying cycles between packages as well as package groups potentially representing layers. DSM (dependency structure matrix) [76], [69] seems to be adapted for such a task but there is no serious empirical experience that validates this claim. From the side of remodularization algorithms, many were defined for procedural languages [57]. However, object-oriented programming languages bring some specific problems linked with late-binding and the fact that a package does not have to be systematically cohesive since it can be an extension of another one [78], [48].

As we are designing and evaluating algorithms and analyses to remodularize applications, we also need a way to understand and assess the results we are obtaining.

Research Agenda. We work on the following items:

Layer identification. We propose an approach to identify layers based on a semi-automatic classification of package and class interrelationships that they contain. However, taking into account the wish or knowledge of the designer or maintainer should be supported.

Cohesion Metric Assessment. We are building a validation framework for cohesion/coupling metrics to determine whether they actually measure what they promise to. We are also compiling a number of traditional metrics for cohesion and coupling quality metrics to evaluate their relevance in a software quality setting.

3.1.3. Software Quality

Research Agenda. Since software quality is fuzzy by definition and a lot of parameters should be taken into account we consider that defining precisely a unique notion of software quality is definitively a Grail in the realm of software engineering. The question is still relevant and important. We work on the two following items:

Quality models. We studied existing quality models and the different options to combine indicators — often, software quality models happily combine metrics, but at the price of losing the explicit relationships between the indicator contributions. There is a need to combine the results of one metric over all the software components of a system, and there is also the need to combine different metric results for any software component. Different combination methods are possible that can give very different results. It is therefore important to understand the characteristics of each method.

Bug prevention. Another aspect of software quality is validating or monitoring the source code to avoid the emergence of well known sources of errors and bugs. We work on how to best identify such common errors, by trying to identify earlier markers of possible errors, or by helping identifying common errors that programmers did in the past.

3.2. Language Constructs for Modular Design

While the previous axis focuses on how to help remodularizing existing software, this second research axis aims at providing new language constructs to build more flexible and recomposable software. We will build on our work on traits [73], [46] and classboxes [35] but also start to work on new areas such as isolation in dynamic languages. We will work on the following points: (1) Traits and (2) Modularization as a support for isolation.

3.2.1. Traits-based program reuse

Context and Problems. Inheritance is well-known and accepted as a mechanism for reuse in object-oriented languages. Unfortunately, due to the coarse granularity of inheritance, it may be difficult to decompose an application into an optimal class hierarchy that maximizes software reuse. Existing schemes based on single inheritance, multiple inheritance, or mixins, all pose numerous problems for reuse.

To overcome these problems, we designed a new composition mechanism called Traits [73], [46]. Traits are pure units of behavior that can be composed to form classes or other traits. The trait composition mechanism is an alternative to multiple or mixin inheritance in which the composer has full control over the trait composition. The result enables more reuse than single inheritance without introducing the drawbacks of multiple or mixin inheritance. Several extensions of the model have been proposed [43], [65], [36], [47] and several type systems were defined [49], [74], [66], [59].

Traits are reusable building blocks that can be explicitly composed to share methods across unrelated class hierarchies. In their original form, traits do not contain state and cannot express visibility control for methods. Two extensions, stateful traits and freezable traits, have been proposed to overcome these limitations. However, these extensions are complex both to use for software developers and to implement for language designers.

Research Agenda: Towards a pure trait language. We plan distinct actions: (1) a large application of traits, (2) assessment of the existing trait models and (3) bootstrapping a pure trait language.

- To evaluate the expressiveness of traits, some hierarchies were refactored, showing code reuse [38]. However, such large refactorings, while valuable, may not exhibit all possible composition problems, since the hierarchies were previously expressed using single inheritance and following certain patterns. We want to redesign from scratch the collection library of Smalltalk (or part of it). Such a redesign should on the one hand demonstrate the added value of traits on a real large and redesigned library and on the other hand foster new ideas for the bootstrapping of a pure trait-based language.

In particular we want to reconsider the different models proposed (stateless [46], stateful [37], and freezable [47]) and their operators. We will compare these models by (1) implementing a trait-based collection hierarchy, (2) analyzing several existing applications that exhibit the need for traits. Traits may be flattened [64]. This is a fundamental property that confers to traits their simplicity and expressiveness over Eiffel's multiple inheritance. Keeping these aspects is one of our priority in forthcoming enhancements of traits.

- Alternative trait models. This work revisits the problem of adding state and visibility control to traits. Rather than extending the original trait model with additional operations, we use a fundamentally different approach by allowing traits to be lexically nested within other modules. This enables traits to express (shared) state and visibility control by hiding variables or methods in their lexical scope. Although the traits' "flattening property" no longer holds when they can be lexically nested, the combination of traits with lexical nesting results in a simple and more expressive trait model. We formally specify the operational semantics of this combination. Lexically nested traits are fully implemented in AmbientTalk, where they are used among others in the development of a Morphic-like UI framework.
- We want to evaluate how inheritance can be replaced by traits to form a new object model. For this purpose we will design a minimal reflective kernel, inspired first from ObjVlisp [42] then from Smalltalk [52].

3.2.2. Reconciling Dynamic Languages and Isolation

Context and Problems. More and more applications require dynamic behavior such as modification of their own execution (often implemented using reflective features [56]). For example, F-script allows one to script Cocoa Mac-OS X applications and Lua is used in Adobe Photoshop. Now in addition more and more applications are updated on the fly, potentially loading untrusted or broken code, which may be problematic for the system if the application is not properly isolated. Bytecode checking and static code analysis are used to enable isolation, but such approaches do not really work in presence of dynamic languages and reflective features. Therefore there is a tension between the need for flexibility and isolation.

Research Agenda: Isolation in dynamic and reflective languages. To solve this tension, we will work on *Sure*, a language where isolation is provided by construction: as an example, if the language does not offer field access and its reflective facilities are controlled, then the possibility to access and modify private data is controlled. In this context, layering and modularizing the meta-level [39], as well as controlling the access to reflective features [40], [41] are important challenges. We plan to:

- Study the isolation abstractions available in erights (http://www.erights.org) [63], [62], and Java's class loader strategies [58], [53].
- Categorize the different reflective features of languages such as CLOS [55], Python and Smalltalk [67] and identify suitable isolation mechanisms and infrastructure [50].
- Assess different isolation models (access rights, capabilities [68]...) and identify the ones adapted to our context as well as different access and right propagation.
- Define a language based on
  - the decomposition and restructuring of the reflective features [39],
  - the use of encapsulation policies as a basis to restrict the interfaces of the controlled objects [72],
  - the definition of method modifiers to support controlling encapsulation in the context of dynamic languages.

An open question is whether, instead of providing restricted interfaces, we could use traits to grant additional behavior to specific instances: without trait application, the instances would only exhibit default public behavior, but with additional traits applied, the instances would get extra behavior. We will develop *Sure*, a modular extension of the reflective kernel of Smalltalk (since it is one of the languages offering the largest set of reflective features such as pointer swapping, class changing, class definition...) [67].

4. Application Domains

4.1. Programming Languages and Tools

Many of the results of RMoD are improving programming languages or development tools for such languages. As such the application domain of these results is as varied as the use of programming languages in general. Pharo, the language that RMoD develops, is used for a very broad range of applications. From pure research experiments to real world industrial use (the Pharo Consortium, http://consortium.pharo.org, has more than 20 company members) Examples are web applications, server backends for mobile applications or even graphical tools and embedded applications

4.2. Software Reengineering

Moose is a language-independent environment for reverse and re-engineering complex software systems. Moose provides a set of services including a common meta-model, metrics evaluation and visualization. As such Moose is used for analysing software systems to support understanding and continous development as well as software quality analysis.

5. Highlights of the Year

5.1. Highlights of the Year

5.1.1. Release of Pharo 5.0

We released a new version Pharo (Pharo 5.0) completly revisited with fundamental changes in the VM (object representation, compiler, ...)

5.1.2. HDR defenses

Anne Etien defended her HDR thesis.

5.1.3. Pharo web for the enterprise

A new book on Pharo.

5.1.4. Guillermo Polito hired as a CNRS engineer

Guillermo Polito a former PhD student in RMod was hired as a CNRS research engineer. This acknolewdges the quality of his research and work.

6. New Software and Platforms

6.1. Pharo

KEYWORDS: Live programmation objet - Reflective system
FUNCTIONAL DESCRIPTION

The platform Pharo is an open-source Smalltalk-inspired language and environment. It provides a platform for innovative development both in industry and research. By providing a stable and small core system, excellent developer tools, and maintained releases, Pharo's goal is to be a platform to build and deploy mission critical applications, while at the same time continue to evolve. In 2016, we released a new version Pharo (Pharo 5.0) completely revisited with fundamental changes in the VM (object representation, compiler, ...)

- Participants: Marcus Denker, Damien Cassou, Stephane Ducasse, Esteban Lorenzano, Damien Pollet, Igor Stasenko, Camillo Bruni, Camille Teruel and Clement Bera
- Partners: BetaNine - Debris publishing - École des Mines de Douai - HR Works - MAD - Pleiad - Sensus - Synectique - Université de Berne - Uqbar foundation Argentina - Vmware - Yesplan
- Contact: Marcus Denker
- URL: http://www.pharo.org

6.2. Moose

FUNCTIONAL DESCRIPTION

Moose is an extensive platform for software and data analysis. It offers multiple services ranging from importing and parsing data, to modeling, to measuring, querying, mining, and to building interactive and visual analysis tools.

- Participants: Stephane Ducasse, Muhammad Bhatti, Andre Cavalcante Hora, Nicolas Anquetil, Anne Etien, Guillaume Larcheveque and Alexandre Bergel
- Partners: Pleiad - Sensus - Synectique - Université de Berne - USI - Vrije Universiteit Brussel - Feenk
- Contact: Stephane Ducasse
- URL: http://www.moosetechnology.org

6.3. Pillar

KEYWORDS: HTML - LaTeX - HTML5
FUNCTIONAL DESCRIPTION

Pillar is a markup syntax and associated tools to write and generate documentation and books. Pillar is currently used to write several books and other documentation. Two platforms have already been created on top of Pillar: PillarHub and Marina.

- Contact: Damien Cassou
- URL: http://www.smalltalkhub.com/#!/~Pier/Pillar

7. New Results

7.1. Practical Validation of Bytecode to Bytecode JIT Compiler Dynamic Deoptimization.

Speculative inlining in just-in-time compilers enables many performance optimizations. However, it also introduces significant complexity. The compiler optimizations themselves, as well as the deoptimization mechanism are complex and error prone. To stabilize our bytecode to bytecode just-in-time compiler, we designed a new approach to validate the correctness of dynamic deoptimization. The approach consists of the symbolic execution of an optimized and an unop-timized bytecode compiled method side by side, deoptimizing the abstract stack at each deoptimization point (where dynamic deoptimization is possible) and comparing the deoptimized and unoptimized abstract stack to detect bugs. The implementation of our approach generated tests for several hundred thousands of methods, which are now available to be run automatically after each commit [13].

7.2. Recording and Replaying System-Specific Conventions.

In other situations, we found that developers sometimes perform sequences of code changes in a systematic way. These sequences consist of small code changes (*e.g.*, create a class, then extract a method to this class), which are applied to groups of related code entities (*e.g.*, some of the methods of a class). We propose to help this task by letting the developer record the sequence of code changes when he first applies it, and then generalize this sequence to apply it in other locations. The evaluation is based on real instances of such sequences that we found in different open source systems. We were able to replay 92% of the examples, which consisted in up to seven code entities modified up to 66 times. We are still working on the approach to allow for (semi-)automatic generalization of the recorded sequence of changes [71], [70].

7.3. Test Case Selection in Industry: an Analysis of Issues Related to Static Approaches

Automatic testing constitutes an important part of everyday development practice. But running all these tests may take hours. This is especially true for large systems involving, for example, the deployment of a web server or communication with a database. For this reason, tests are not launched as often as they should be and are mostly run at night. The company wishes to improve its development and testing process by giving to developers rapid feedback after a change. An interesting solution to give developers rapid feedback after a change is to reduce the number of tests to run by identifying only those exercising the piece of code changed. Two main approaches are proposed in the literature: static and dynamic. We evaluate these approaches on three industrial, closed source, cases to understand the strengths and weaknesses of each solution. We also propose a classification of problems that may arise when trying to identify the tests that cover a method.

8. Bilateral Contracts and Grants with Industry

8.1. Bilateral Grants with Industry

8.1.1. BlockChain

We started a new collaboration with a local company about tools and languages in the context of Blockchain systems. The collaboration started with a 2 month exploration phase involving an engineer at Inria Tech. A postdoc or PhD will start in 2017.

8.1.2. Worldline CIFRE

We are working on improving the testing behaviour of the developers. The PhD started in October 2014 and is ongoing.

8.1.3. Thales CIFRE

We are working on large industrial project rearchitecturization. The PhD started in January 2015 and is ongoing.

8.1.4. Pharo Consortium

The Pharo Consortium was founded in 2012 and is growing constantly. As of end 2016, it has 23 company members, 12 academic partners and 3 sponsoring companies. Inria supports the consortium with one full time engineer starting in 2011. More at http://consortium.pharo.org.

9. Partnerships and Cooperations

9.1. Regional Initiatives

Participants: Anne Etien, Nicolas Anquetil, Olivier Auverlot, Stéphane Ducasse. From Sept 2016 to Dec. 2018.

Lille Nord Europe European Associated Team with the PreCISE research center of Pr. A. Cleve from Namur University (Belgium).

This project aims to study the co-evolution between database structure and programs and to propose recommendations to perform required changes on cascade. These programs are either internal to the schema as functions or triggers or external as applications written in Java or Php built on top of the DB. Our intuition is that software engineering techniques can be efficient for such issues. This project also aims to unify the abstract representation of the DB and its relationships with the internal or external program.

9.2. European Initiatives

9.2.1. Collaborations with Major European Organizations

Marco Naddéo is a PhD student co-supervised by Damien Cassou, Stéphane Ducasse and Viviana Bono from University of Turin (Italy).

9.3. International Initiatives

9.3.1. Inria International Labs

Inria Chile
Associate Team involved in the International Lab:

9.3.1.1. PLOMO2

Title: Infrastructure for a new generation of development tools

International Partner (Institution - Laboratory - Researcher):

Universidad de Chile (Chile) - Computer Science Department, PLEIAD laboratory (DCC) - Alexander Bergel

Start year: 2014

See also: http://pleiad.cl/research/plomo2

Performing effective software development and maintenance are best achieved with effective tool support. Provided by a variety of tools, each one presenting a specific kinds of information supporting the task at hand. With Plomo2, we want to invent a new generation tools to navigate and profile programs by combining dynamic information with visualization to improve the development environment.

9.3.2. Inria Associate Teams Not Involved in an Inria International Labs

9.3.2.1. Informal International Partners

We are working with the Uqbar team from different argentinian universities. We hired three of the people: Nicolas Passerini(engineer), Esteban Lorenzano (engineer) and Pablo Tesone (PhD).

We are starting to work with Dr. Robert Pergl from the University of Prague.

9.3.3. Participation in Other International Programs

9.3.3.1. STIC AmSud projects

We were involved in two STIC AmSud projects:

Participants: Damien Cassou [correspondant], Gustavo Santos [RMoD], Martin Dias [RMoD], David Röthlisberger [UDP - Universidad Diego Portales, Santiago, Chile], Marcelo Almeida Maia [UFU - Federal University of Uberlândia, Brasil], Romain Robbes [Departamento de Ciencias de la Computación (DCC), Universidad de Chile, Santiago, Chile], Martin Monperrus [Spirals]. Project Partners: Inria RMOD, Inria Spirals, DCC Universidad de Chile, Universidad Diego Portale Chile, Federal University of Uberlândia, Brasil.

This project aims at facilitating the usage of frameworks and application programming interfaces (APIs) by mining software repositories. Our intuition is that mining reveals how existing projects instantiate these frameworks. By locating concrete framework instantiations in existing projects, we can recommend to developers the concrete procedures for how to use a particular framework for a particular task in a new system. Our project also tackles the challenge of adapting existing systems to new versions of a framework or API by seeking repositories for how other systems adapted to such changes. We plan to integrate recommendations of how to instantiate a framework and adapt to changes directly in the development environment. Those points taken together, considerably distinguish our approach from existing research in the area of framework engineering.

Thanks to this project, a PhD student of Federal University of Uberlândia in Brasil (Klérisson Vinícius Ribeiro da Paixão) did a six months internship in RMod, and prof. Marcelo Almeida Maia (from the same university) visited us for one week.

Participants: Nicolas Anquetil [correspondant], Anne Etien [RMoD], Gustavo Santos [RMoD], Marco Tulio Valente [UFMG - Federal University of Minas Gerais, Brazil], Alexander Bergel [Departamento de Ciencias de la Computación (DCC), Universidad de Chile, Santiago, Chile], Project Partners: Inria RMOD, DCC Universidad de Chile, Federal University of Minas Gerais, Brazil.

The goals of the collaboration is to provide tools to help software engineer restructure a large software system in an iterative and incremental way with input from both expert architect and advanced tool. The tools consist of: extraction of a model from source code, manipulation of the model to experiment with possible restructuring, architecture evaluation tool, recommendation tool to help the software engineers to define the new structure, and tool to back port the modification to the source code.

Pr. Marco Tulio Valente of Federal University of Minas Gerais (Brazil)visited RMod for one week.

9.4. International Research Visitors

9.4.1. Visits of International Scientists

- Prof. Serge Demeyer, 2 days
- Prof. Marcelo Almeida Maia, 1 week, November 2016;
- Prof. Alain Plantec, 5 days, November 2016;
- Prof. Alexandre Bergel, 1 week, December 2016;
- Prof. Marco Tulio Valente, 1 week, December 2016;
- Eliot Miranda, Cadence,1 week, August 2016;
- Dr. Andrei Chis, 3 days, Decembrer 2016.

9.4.1.1. Internships

- Klérisson Vinícius Ribeiro da Paixão, Federal University of Uberlândia, Uberlândia (MG), Brazil, from September, 2015 to July, 2016;
- Lionel Akue, University of Lomé, Togo, from Oct 2016 to Nov 2016
- Julien Delplanque, University of Mons, Belgium, from September 2016 to December 2016.

9.4.2. Visits to International Teams

The whole RMod team spent two days in January at the Vrije Universiteit Brussel (Belgium) to discuss with the Software Languages Lab team: Coen De Roover, Elisa Gonzalez Boix, and their students.

Stéphane Ducasse and Damien Cassou visited Dr. Robert Pergl's team at Czech Technical University in Prague, one week in April.

9.4.2.1. Research Stays Abroad

Marcus Denker: Visit PLEIAD DCC University of Chile, Santiago de Chile 04/12-21/12. Visit in the context of the Inria Associated Team PLOMO2.

10. Dissemination

10.1. Promoting Scientific Activities

10.1.1. Scientific Events Selection

10.1.1.1. Chair of Conference Program Committees

- Anne Etien acted as Program chair for the following international events: IWST16 (International Worshop on SmallTalk Technologies) and VISSOFT 2016 (IEEE Working Conference on Software Visualization): NIER and Tool Track
- Stéphane Ducasse acted as Program Chair for ESUG 2016.

10.1.1.2. Member of the Conference Program Committees

- Stéphane Ducasse
  - International Conference 18th International Conference on Fundamental Approaches to Software Engineering (FASE) 2016.
  - For health reasons, I declined the PC participation to ICSE 2013, ECOOP 2013, ECOOP 2014, ECOOP 2015 (but I was the workshop chair) ECOOP 2016.
  - IWST'16 (International Worshop on SmallTalk Technologies).
- Anne Etien
  - DChanges'16 (4th International Workshop on Document Changes: Modeling, Detection, Storage and Visualization);
  - ME'16 (10th Workshop on Models and Evolution);
  - Modelwards'16 (4th International Conference on Model-Driven Engineering and Software Development).
  - IWST'16 (International Worshop on SmallTalk Technologies).
- Nicolas Anquetil
  - ICSOFT-EA'16 (11th International Conference on Software Technologies and Applications);
  - IWST'16 (International Worshop on SmallTalk Technologies).
- Marcus Denker
  - SANER 2017 (23rd IEEE International Conference on Software Analysis, Evolution, and Reengineering)
  - Meta 2016 (International Workshop, SPLASH 16)
  - SLE 2016 (International Conference on Software Language Engineering)
  - DLS 2016 (Dynamic Languages Symposium at SPLASH)
  - IWST'16 (International Worshop on SmallTalk Technologies).
- Clément Béra
  - ICOOOLPS'16 (11th Implementation, Compilation, Optimization of Object-Oriented Languages, Programs and Systems Workshop)
  - IWST'16 (International Worshop on SmallTalk Technologies).

10.1.2. Journal

10.1.2.1. Reviewer - Reviewing Activities

- Anne Etien reviewed manuscripts for the Software Quality Journal

- Nicolas Anquetil reviewed manuscripts for the ACM Transactions on Software Engineering and Methodology (TOSEM); Journal of Software: Evolution and Process (JSME); Computers & Operations Research (COR).
- Stéphane Ducasse reviewed manuscripts for Information ; Journal of Software: Evolution and Process (JSME); Journal of Science of Computer Programming; Journal of Object Technologies (JOT).
- Stéphane Ducasse and Marcus Denker are Moderators for arXiv cs.SE. http://arxiv.org/list/cs.SE/recent

10.1.3. Invited Talks

- Stéphane Ducasse gave a presentation at CITI laboratory Lyon, Diverse Team seminar Inria Rennes, La Sorbonne
- Clément Béra gave a presentation "Sista: Speculative inlining, Smalltalk-style", with Eliot Miranda (Cadence Design Systems) at the Stanford EE Computer Systems Colloquium, Stanford University (https://www.youtube.com/watch?v=f4Cvia-HZ-w).

10.1.4. Scientific Expertise

- Anne Etien reviewed several proposals for *Jeune Entreprise Innovante* and *Crédit Impôt Recherche*;
- Nicolas Anquetil acted as expert to evaluate project proposal for *Fondo Nacional de Desarrollo Cientifico y Tecnologico* (FONDECYT), Chile;
- Marcus Denker reviewed a research proposal for FONDECYT (Chilean National Science and Technology Commission);
- Damien Pollet reviewed projects proposal for NWO (Netherlands Organisation for Scientific Research) Veni grant proposal;
- Stéphane Ducasse participated to FWO (Belgium) PhD grant panel;
- Damien Cassou was a member of the Apps Togo jury, a contest from Ministry of Industry of Togo to encourage innovation in apps development (http://www.republicoftogo.com/Toutes-les-rubriques/Tech-Web/Voici-les-meilleures-applis-du-moment).

10.2. Teaching - Supervision - Juries

10.2.1. Teaching

Master : C. Demarey, Intégration Continue, 4h, M2, GIS2A5, Polytech-Lille, France

Master : C. Demarey, Intégration Continue, 12h, M2, IAGL, Université de Lille 1, France

Master: Anne Etien, Metamodelisation for Reverse Engineering, 25h, M2, Montpellier, France

Master: Anne Etien, Test et Maintenance, 10h, M2, Polytech-Lille, France

Master: Anne Etien, Test et Maintenance, 14h, M2, Polytech-Lille, France

Master : Anne Etien, Système d'information objet, 22h, M1, Polytech-Lille, France

Master : Anne Etien, Bases de données, 44h, M1, Polytech-Lille, France

Master : Anne Etien, Bases de données Avancées, 12h, M1, Polytech-Lille, France

Licence : Anne Etien, Bases de données, 22h, L3, Polytech-Lille, France

Licence : Nicolas Anquetil, Conception et programmation objet avancées, 52h, L2, Univ. Lille1, IUT-A, France

Licence : Nicolas Anquetil, Principes des systèmes d'exploitation, 40h, L2, Univ. Lille1, IUT-A, France

Licence : Nicolas Anquetil, Modélisation mathématique, 14h, L2, univ. Lille1, IUT-A, France

Licence : Nicolas Anquetil, Conception et développement d'applications mobiles, 30h, L2, univ. Lille1, IUT-A, France

Licence : Nicolas Anquetil, Méthodologie de la production d'applications, 33h, L2, univ. Lille1, IUT-A, France

Licence : Damien Pollet, Java lecture, 30h, L3, Telecom Lille, France

Licence : Damien Pollet, Java lecture, 20h, L3, Telecom Lille, France

Licence : Damien Pollet, Java project, 20h, L3, Telecom Lille, France

Licence : Damien Pollet, Computer architecture,10h, L3, Telecom Lille, France

Master : Damien Pollet, Technologies for information systems, 30h, M1, Telecom Lille, France

Master : Damien Pollet, Network algorithms, 30h, M2, Telecom Lille, France

Master : Damien Pollet, Software engineering, 6h, M1, Telecom Lille, France

Licence: Damien Cassou, Algorithmique et programmation, 25h, L1, Polytech-Lille, France

Licence: Damien Cassou, OpenDevs—Communauté de Développeurs, 30h, L3, Univ. Lille-1, France

Licence : Clément Béra, Algorithmique et programmation, 35h, L1, Polytech-Lille, France

Licence : Clément Béra, Conception orienté objet, 35h, L1, Univ. Lille-1, France

Master : Clément Béra, Génie Logiciel, 40h, M3, Univ. Lille-1, France

Master : Vincent Blondeau, Bases de données, 12h, M1, Polytech-Lille, France

E-learning

Mooc: Stéphane Ducasse, Damien Cassou, Luc Fabresse, "Programmation Objet Immersive en Pharo / Live Object Programming in Pharo", 7 weeks, by France Université Numerique and Inria, for programmers, 3400 participants

10.2.2. Supervision

PhD in progress : Vincent Blondeau, "Studying and Enhancing Testing Habits of Developers in a Large IT Company", CIFRE Worldline, started Oct 2014, Anne Etien, Nicolas Anquetil.

PhD in progress : Brice Govin, "Support to implement a rejuvenated software architecture in legacy software", CIFRE Thales, started Jan 2015, Anne Etien, Nicolas Anquetil, Stéphane Ducasse.

PhD in progress : Gustavo Santos, "Assessing and Improving Code Transformations to Support Software Evolution", started April 2014, Anne Etien, Nicolas Anquetil.

PhD: Camille Teruel, "Security for dynamic languages", 21/01/2016, Stéphane Ducasse, Damien Cassou

PhD in progress: Marco Naddéo, "Evolvable and reusable software: from language design to developer tools", started 01/01/2014, Viviana Bono (University of Turin), Damien Cassou, Stéphane Ducasse

PhD in progress: Clément Béra, "Evolvable Runtimes for a Changing World", started Oct 2014,Stéphane Ducasse, Marcus Denker.

PhD in progresss: Klérisson Paixao, "Automating reactions for API changes based on dynamic analysis", started 16/08/2014, Marcelo Maia, Damien Cassou, Nicolas Anquetil.

10.2.3. Juries

- Anne Etien: Jonathan Pepin, PhD thesis, "Architecture d'entreprise : alignement des cartographies me´tiers et applicatives du syste`me d'informations", Dec. 5th, 2016, University of Nantes (France).
- Nicolas Anquetil: David Michael Cutting, PhD thesis, "Enhancing Legacy Software System Analysis by Combining Behavioural and Semantic Information Services", Aug. 5th, 2016, University of East Anglia, Norwich (England).

10.3. Popularization

- Stéphane Ducasse and Damien Cassou were important contributors of the “Enterprise Pharo” book (http://files.pharo.org/books/enterprise-pharo/).
- Brice Govin and Vincent Blondeau participated in the contest “My thesis in 180 seconds”

11. Bibliography

Major publications by the team in recent years

[1] N. ANQUETIL, K. M. DE OLIVEIRA, K. D. DE SOUSA, M. G. BATISTA DIAS. *Software maintenance seen as a knowledge management issue*, in "Information Software Technology", 2007, vol. 49, n^{o} 5, pp. 515–529 [*DOI :* 10.1016/J.INFSOF.2006.07.007], http://rmod.lille.inria.fr/archives/papers/Anqu07a-IST-MaintenanceKnowledge.pdf

[2] M. DENKER, S. DUCASSE, É. TANTER. *Runtime Bytecode Transformation for Smalltalk*, in "Journal of Computer Languages, Systems and Structures", July 2006, vol. 32, n^{o} 2-3, pp. 125–139 [*DOI :* 10.1016/J.CL.2005.10.002], http://rmod.lille.inria.fr/archives/papers/Denk06a-COMLAN-RuntimeByteCode.pdf

[3] S. DUCASSE, O. NIERSTRASZ, N. SCHÄRLI, R. WUYTS, A. P. BLACK. *Traits: A Mechanism for fine-grained Reuse*, in "ACM Transactions on Programming Languages and Systems (TOPLAS)", March 2006, vol. 28, n^{o} 2, pp. 331–388 [*DOI :* 10.1145/1119479.1119483], http://scg.unibe.ch/archive/papers/Duca06bTOPLASTraits.pdf

[4] S. DUCASSE, D. POLLET. *Software Architecture Reconstruction: A Process-Oriented Taxonomy*, in "IEEE Transactions on Software Engineering", July 2009, vol. 35, n^{o} 4, pp. 573-591 [*DOI :* 10.1109/TSE.2009.19], http://rmod.lille.inria.fr/archives/papers/Duca09c-TSE-SOAArchitectureExtraction.pdf

[5] S. DUCASSE, D. POLLET, M. SUEN, H. ABDEEN, I. ALLOUI. *Package Surface Blueprints: Visually Supporting the Understanding of Package Relationships*, in "ICSM'07: Proceedings of the IEEE International Conference on Software Maintenance", 2007, pp. 94–103, http://scg.unibe.ch/archive/papers/Duca07cPackageBlueprintICSM2007.pdf

[6] A. KUHN, S. DUCASSE, T. GÎRBA. *Semantic Clustering: Identifying Topics in Source Code*, in "Information and Software Technology", March 2007, vol. 49, n^{o} 3, pp. 230–243 [*DOI :* 10.1016/J.INFSOF.2006.10.017], http://scg.unibe.ch/archive/drafts/Kuhn06bSemanticClustering.pdf

[7] J. LAVAL, S. DENIER, S. DUCASSE, A. BERGEL. *Identifying cycle causes with Enriched Dependency Structural Matrix*, in "WCRE '09: Proceedings of the 2009 16th Working Conference on Reverse Engineering", Lille, France, 2009, http://rmod.lille.inria.fr/archives/papers/Lava09c-WCRE2009-eDSM.pdf

[8] O. NIERSTRASZ, S. DUCASSE, T. GÎRBA. *The Story of Moose: an Agile Reengineering Environment*, in "Proceedings of the European Software Engineering Conference", New York NY, M. WERMELINGER, H. GALL (editors), ESEC/FSE'05, ACM Press, 2005, pp. 1–10, Invited paper [*DOI :* 10.1145/1095430.1081707], http://scg.unibe.ch/archive/papers/Nier05cStoryOfMoose.pdf

[9] J. SINGER, T. LETHBRIDGE, N. VINSON, N. ANQUETIL. *An examination of software engineering work practices*, in "Proceedings of the 1997 conference of the Centre for Advanced Studies on Collaborative research", CASCON '97, IBM Press, 1997, http://rmod.lille.inria.fr/archives/papers/Sing97a-SoftwareEngineeringWorkPractices.pdf

[10] S. C. B. DE SOUZA, N. ANQUETIL, K. M. DE OLIVEIRA. *A study of the documentation essential to software maintenance*, in "Proceedings of the 23rd annual international conference on Design of communication: documenting & designing for pervasive information", New York, NY, USA, SIGDOC '05, ACM, 2005, pp. 68–75, http://dx.doi.org/10.1145/1085313.1085331

Publications of the year

Doctoral Dissertations and Habilitation Theses

[11] A. ETIEN. *Metamodelisation to support Test and Evolution*, Université de Lille, June 2016, Habilitation à diriger des recherches, https://hal.inria.fr/tel-01352817

[12] C. TERUEL. *Adaptability and Encapsulation in Dynamically-Typed Languages: Taming Reflection and Extension Methods*, Université des Sciences et Technologies de Lille, January 2016, https://tel.archives-ouvertes.fr/tel-01290364

Articles in International Peer-Reviewed Journals

[13] C. BERA, E. MIRANDA, M. DENKER, S. DUCASSE. *Practical Validation of Bytecode to Bytecode JIT Compiler Dynamic Deoptimization*, in "Journal of Object Technology (JOT)", April 2016, vol. 15, n° 2, pp. 1:1-26 [*DOI :* 10.5381/JOT.2016.15.2.A1], https://hal.inria.fr/hal-01299371

[14] V. BLONDEAU, A. ETIEN, N. ANQUETIL, S. CRESSON, P. CROISY, S. DUCASSE. *Test Case Selection in Industry: an Analysis of Issues Related to Static Approaches*, in "Software Quality Journal", 2016 [*DOI :* 10.1007/S11219-016-9328-4], https://hal.inria.fr/hal-01344842

[15] J. DE KOSTER, S. MARR, T. VAN CUTSEM, T. D 'HONDT. *Domains: Sharing State in the Communicating Event-Loop Actor Model*, in "Computer Languages, Systems and Structures", January 2016 [*DOI :* 10.1016/J.CL.2016.01.003], https://hal.inria.fr/hal-01273665

[16] M. DE WAEL, S. MARR, B. DE FRAINE, T. VAN CUTSEM, W. DE MEUTER. *Partitioned Global Address Space Languages*, in "ACM Computing Surveys", January 2016, 29 p. , https://hal.inria.fr/hal-01109405

[17] A. HORA, R. ROBBES, M. TULIO VALENTE, N. ANQUETIL, A. ETIEN, S. DUCASSE. *How do Developers React to API Evolution? a Large-Scale Empirical Study*, in "Software Quality Journal", October 2016 [*DOI :* 10.1007/S11219-016-9344-4], https://hal.inria.fr/hal-01417930

[18] C. MAFFORT, M. T. VALENTE, R. TERRA, M. BIGONHA, N. ANQUETIL, A. HORA. *Mining Architectural Violations from Version History*, in "Empirical Software Engineering", June 2016, vol. 21, n° 3, 41 p. [*DOI :* 10.1007/S10664-014-9348-2], https://hal.inria.fr/hal-01075642

[19] M. SMID, F. G. RODRÍGUEZ-GONZÁLEZ, A. M. SIEUWERTS, R. SALGADO, W. J. C. PRAGER-VAN DER SMISSEN, M. V. D. VLUGT-DAANE, A. VAN GALEN, S. NIK-ZAINAL, J. STAAF, A. B. BRINKMAN, M. J. VAN DE VIJVER, A. L. RICHARDSON, A. FATIMA, K. BERENTSEN, A. BUTLER, S. MARTIN, H. R. DAVIES, R. DEBETS, M. E. M.-V. GELDER, C. H. M. VAN DEURZEN, G. MACGROGAN, G. G. G. M. VAN

DEN EYNDEN, C. PURDIE, A. M. THOMPSON, C. CALDAS, P. N. SPAN, P. T. SIMPSON, S. R. LAKHANI, S. VAN LAERE, C. DESMEDT, M. RINGNÉR, S. TOMMASI, J. EYFORD, A. BROEKS, A. VINCENT-SALOMON, P. A. FUTREAL, S. KNAPPSKOG, T. A. KING, G. THOMAS, A. VIARI, A. LANGERØD, A.-L. BØRRESEN-DALE, E. BIRNEY, H. G. STUNNENBERG, M. STRATTON, J. A. FOEKENS, J. W. M. MARTENS. *Breast cancer genome and transcriptome integration implicates specific mutational signatures with immune cell infiltration*, in "Nature Communications", 2016, vol. 7 [*DOI :* 10.1038/NCOMMS12910], https://hal.inria.fr/hal-01388445

International Conferences with Proceedings

[20] A. CAVALCANTE HORA, M. TULIO VALENTE, R. ROBBES, N. ANQUETIL. *When should internal interfaces be promoted to public?*, in "FSE 2016 Proceedings of the 2016 24th ACM SIGSOFT International Symposium on Foundations of Software Engineering", Seattle, United States, November 2016 [*DOI :* 10.1145/2950290.2950306], https://hal.inria.fr/hal-01417888

[21] J. P. SANDOVAL ALCOCER, M. DENKER, A. BERGEL, Y. ACURANA. *Dynamically Composing Collection Operations through Collection Promises*, in "Proceedings of the 11th Edition of the International Workshop on Smalltalk Technologies", Prague, Czech Republic, Proceedings of the 11th Edition of the International Workshop on Smalltalk Technologies, ESUG, August 2016, pp. 8:1–8:5 [*DOI :* 10.1145/2991041.2991049], https://hal.inria.fr/hal-01358347

Conferences without Proceedings

[22] T. ARLOING, Y. DUBOIS, D. CASSOU, S. DUCASSE. *Pillar: A Versatile and Extensible Lightweight Markup Language*, in "International Workshop on Smalltalk Technologies", Prague, Czech Republic, August 2016, https://hal.inria.fr/hal-01353882

[23] C. BERA. *A low Overhead Per Object Write Barrier for the Cog VM*, in "IWST 16", Pragues, Czech Republic, 2016, https://hal.archives-ouvertes.fr/hal-01356338

[24] V. BLONDEAU, N. ANQUETIL, S. DUCASSE, S. CRESSON, P. CROISY. *Test Selection with Moose In Industry*, in "IWST'16", Prague, Czech Republic, August 2016, https://hal.inria.fr/hal-01352468

[25] S. DUCASSE, E. MIRANDA, A. PLANTEC. *Pragmas: Literal Messages as Powerful Method Annotations*, in "International Workshop on Smalltalk Technologies - IWST 2016", Prague, Czech Republic, August 2016, https://hal.inria.fr/hal-01353592

[26] B. GOVIN, A. MONEGIER DU SORBIER, N. ANQUETIL, S. DUCASSE. *Clustering technique for conceptual clusters*, in "IWST'16 International Workshop on Smalltalk Technologies", Prague, Czech Republic, August 2016, https://hal.archives-ouvertes.fr/hal-01353205

[27] N. MILOJKOVIĆ, C. BERA, M. GHAFARI, O. NIERSTRASZ. *Inferring Types by Mining Class Usage Frequency from Inline Caches*, in "IWST'16", Pragues, Czech Republic, 2016, https://hal.archives-ouvertes.fr/hal-01357071

[28] M. RIZUN, G. SANTOS, S. DUCASSE, C. TERUEL. *Phorms: Pattern Combinator Library for Pharo*, in "International Workshop on Smalltalk Technologies", Prague, Czech Republic, August 2016, https://hal.inria.fr/hal-01353883

[29] R. SALGADO, S. DUCASSE. *Lowcode: Extending Pharo with C Types to Improve Performance*, in "International Workshop on Smalltalk Technologies", Prague, Czech Republic, August 2016, https://hal.inria.fr/hal-01353884

Research Reports

[30] A. BERGEL, S. DUCASSE, M. DENKER, J. FABRY. *PLOMO2 Associate Team Final Report*, Inria, October 2016, https://hal.inria.fr/hal-01389983

[31] M. DENKER, N. ANQUETIL, D. CASSOU, S. DUCASSE, A. ETIEN, D. POLLET. *Project-Team RMoD 2015 Activity Report*, Inria Lille - Nord Europe, February 2016, https://hal.inria.fr/hal-01267026

Scientific Popularization

[32] O. AUVERLOT, S. DUCASSE. *Un Chat en Pharo*, January 2016, Article in GNU/Linux Magazine, two parts http://connect.ed-diamond.com/GNU-Linux-Magazine/GLMF-189/Un-chat-en-Pharo-le-serveur http://connect.ed-diamond.com/GNU-Linux-Magazine/GLMF-189/Un-chat-en-Pharo-le-client2, https://hal.inria.fr/hal-01353594

Other Publications

[33] N. PAPOULIAS, M. DENKER, S. DUCASSE, L. FABRESSE. *End-User Abstractions for Meta-Control: Reifying the Reflectogram*, January 2017, working paper or preprint, https://hal.inria.fr/hal-01424787

References in notes

[34] N. ANQUETIL. *A Comparison of Graphs of Concept for Reverse Engineering*, in "Proceedings of the 8th International Workshop on Program Comprehension", Washington, DC, USA, IWPC'00, IEEE Computer Society, 2000, pp. 231–, http://rmod.lille.inria.fr/archives/papers/Anqu00b-ICSM-GraphsConcepts.pdf

[35] A. BERGEL, S. DUCASSE, O. NIERSTRASZ. *Classbox/J: Controlling the Scope of Change in Java*, in "Proceedings of 20th International Conference on Object-Oriented Programming, Systems, Languages, and Applications (OOPSLA'05)", New York, NY, USA, ACM Press, 2005, pp. 177–189 [*DOI :* 10.1145/1094811.1094826], http://scg.unibe.ch/archive/papers/Berg05bclassboxjOOPSLA.pdf

[36] A. BERGEL, S. DUCASSE, O. NIERSTRASZ, R. WUYTS. *Stateful Traits*, in "Advances in Smalltalk — Proceedings of 14th International Smalltalk Conference (ISC 2006)", LNCS, Springer, August 2007, vol. 4406, pp. 66–90, http://dx.doi.org/10.1007/978-3-540-71836-9_3

[37] A. BERGEL, S. DUCASSE, O. NIERSTRASZ, R. WUYTS. *Stateful Traits and their Formalization*, in "Journal of Computer Languages, Systems and Structures", 2008, vol. 34, n^{o} 2-3, pp. 83–108, http://dx.doi.org/10.1016/j.cl.2007.05.003

[38] A. P. BLACK, N. SCHÄRLI, S. DUCASSE. *Applying Traits to the Smalltalk Collection Hierarchy*, in "Proceedings of 17th International Conference on Object-Oriented Programming Systems, Languages and Applications (OOPSLA'03)", October 2003, vol. 38, pp. 47–64 [*DOI :* 10.1145/949305.949311], http://scg.unibe.ch/archive/papers/Blac03aTraitsHierarchy.pdf

[39] G. BRACHA, D. UNGAR. *Mirrors: design principles for meta-level facilities of object-oriented programming languages*, in "Proceedings of the International Conference on Object-Oriented Programming, Systems,

Languages, and Applications (OOPSLA'04), ACM SIGPLAN Notices", New York, NY, USA, ACM Press, 2004, pp. 331–344, http://bracha.org/mirrors.pdf

[40] D. CAROMEL, J. VAYSSIÈRE. *Reflections on MOPs, Components, and Java Security*, in "ECOOP '01: Proceedings of the 15th European Conference on Object-Oriented Programming", Springer-Verlag, 2001, pp. 256–274

[41] D. CAROMEL, J. VAYSSIÈRE. *A security framework for reflective Java applications*, in "Software: Practice and Experience", 2003, vol. 33, n° 9, pp. 821–846, http://dx.doi.org/10.1002/spe.528

[42] P. COINTE. *Metaclasses are First Class: the ObjVlisp Model*, in "Proceedings OOPSLA '87, ACM SIGPLAN Notices", December 1987, vol. 22, pp. 156–167

[43] S. DENIER. *Traits Programming with AspectJ*, in "Actes de la Première Journée Francophone sur le Développement du Logiciel par Aspects (JFDLPA'04)", Paris, France, P. COINTE (editor), September 2004, pp. 62–78

[44] S. DUCASSE, T. GÎRBA. *Using Smalltalk as a Reflective Executable Meta-Language*, in "International Conference on Model Driven Engineering Languages and Systems (Models/UML 2006)", Berlin, Germany, LNCS, Springer-Verlag, 2006, vol. 4199, pp. 604–618 [*DOI :* 10.1007/11880240_42], http://scg.unibe.ch/archive/papers/Duca06dMOOSEMODELS2006.pdf

[45] S. DUCASSE, T. GÎRBA, M. LANZA, S. DEMEYER. *Moose: a Collaborative and Extensible Reengineering Environment*, in "Tools for Software Maintenance and Reengineering", Milano, RCOST / Software Technology Series, Franco Angeli, 2005, pp. 55–71, http://scg.unibe.ch/archive/papers/Duca05aMooseBookChapter.pdf

[46] S. DUCASSE, O. NIERSTRASZ, N. SCHÄRLI, R. WUYTS, A. P. BLACK. *Traits: A Mechanism for fine-grained Reuse*, in "ACM Transactions on Programming Languages and Systems (TOPLAS)", March 2006, vol. 28, n° 2, pp. 331–388 [*DOI :* 10.1145/1119479.1119483], http://scg.unibe.ch/archive/papers/Duca06bTOPLASTraits.pdf

[47] S. DUCASSE, R. WUYTS, A. BERGEL, O. NIERSTRASZ. *User-Changeable Visibility: Resolving Unanticipated Name Clashes in Traits*, in "Proceedings of 22nd International Conference on Object-Oriented Programming, Systems, Languages, and Applications (OOPSLA'07)", New York, NY, USA, ACM Press, October 2007, pp. 171–190 [*DOI :* 10.1145/1297027.1297040], http://scg.unibe.ch/archive/papers/Duca07b-FreezableTrait.pdf

[48] A. DUNSMORE, M. ROPER, M. WOOD. *Object-Oriented Inspection in the Face of Delocalisation*, in "Proceedings of ICSE '00 (22nd International Conference on Software Engineering)", ACM Press, 2000, pp. 467–476

[49] K. FISHER, J. REPPY. *Statically typed traits*, University of Chicago, Department of Computer Science, December 2003, n° TR-2003-13, http://www.cs.uchicago.edu/research/publications/techreports/TR-2003-13

[50] P. W. L. FONG, C. ZHANG. *Capabilities as alias control: Secure cooperation in dynamically extensible systems*, Department of Computer Science, University of Regina, 2004

[51] M. FURR, J.-H. AN, J. S. FOSTER. *Profile-guided static typing for dynamic scripting languages*, in "OOPSLA'09", 2009

[52] A. GOLDBERG. *Smalltalk 80: the Interactive Programming Environment*, Addison Wesley, Reading, Mass., 1984

[53] L. GONG. *New security architectural directions for Java*, in "compcon", 1997, vol. 0, 97 p. , http://dx.doi.org/10.1109/CMPCON.1997.584679

[54] M. HICKS, S. NETTLES. *Dynamic software updating*, in "ACM Transactions on Programming Languages and Systems", nov 2005, vol. 27, n^{o} 6, pp. 1049–1096, http://dx.doi.org/10.1145/1108970.1108971

[55] G. KICZALES, J. DES RIVIÈRES, D. G. BOBROW. *The Art of the Metaobject Protocol*, MIT Press, 1991

[56] G. KICZALES, L. RODRIGUEZ. *Efficient Method Dispatch in PCL*, in "Proceedings of ACM conference on Lisp and Functional Programming", Nice, 1990, pp. 99–105

[57] R. KOSCHKE. *Atomic Architectural Component Recovery for Program Understanding and Evolution*, Universität Stuttgart, 2000, http://www2.informatik.uni-stuttgart.de/cgi-bin/NCSTRL/NCSTRL_view.pl?id=DIS-2000-05&mod=0&engl=0&inst=PS

[58] S. LIANG, G. BRACHA. *Dynamic Class Loading in the Java Virtual Machine*, in "Proceedings of OOPSLA '98, ACM SIGPLAN Notices", 1998, pp. 36–44

[59] L. LIQUORI, A. SPIWACK. *FeatherTrait: A Modest Extension of Featherweight Java*, in "ACM Transactions on Programming Languages and Systems (TOPLAS)", 2008, vol. 30, n^{o} 2, pp. 1–32 [*DOI : 10.1145/1330017.1330022*], http://www-sop.inria.fr/members/Luigi.Liquori/PAPERS/toplas-07.pdf

[60] B. LIVSHITS, T. ZIMMERMANN. *DynaMine: finding common error patterns by mining software revision histories*, in "SIGSOFT Software Engineering Notes", September 2005, vol. 30, n^{o} 5, pp. 296-305

[61] R. C. MARTIN. *Agile Software Development. Principles, Patterns, and Practices*, Prentice-Hall, 2002

[62] M. S. MILLER. *Robust Composition: Towards a Unified Approach to Access Control and Concurrency Control*, Johns Hopkins University, Baltimore, Maryland, USA, May 2006

[63] M. S. MILLER, C. MORNINGSTAR, B. FRANTZ. *Capability-based Financial Instruments*, in "FC '00: Proceedings of the 4th International Conference on Financial Cryptography", Springer-Verlag, 2001, vol. 1962, pp. 349–378

[64] O. NIERSTRASZ, S. DUCASSE, N. SCHÄRLI. *Flattening Traits*, in "Journal of Object Technology", May 2006, vol. 5, n^{o} 4, pp. 129–148, http://www.jot.fm/issues/issue_2006_05/article4

[65] P. J. QUITSLUND. *Java Traits — Improving Opportunities for Reuse*, OGI School of Science & Engineering, Beaverton, Oregon, USA, September 2004, n^{o} CSE-04-005

[66] J. REPPY, A. TURON. *A Foundation for Trait-based Metaprogramming*, in "International Workshop on Foundations and Developments of Object-Oriented Languages", 2006

[67] F. RIVARD. *Pour un lien d'instanciation dynamique dans les langages à classes*, in "JFLA96", Inria — collection didactique, January 1996

[68] J. H. SALTZER, M. D. SCHOROEDER. *The Protection of Information in Computer Systems*, in "Fourth ACM Symposium on Operating System Principles", IEEE, September 1975, vol. 63, pp. 1278–1308

[69] N. SANGAL, E. JORDAN, V. SINHA, D. JACKSON. *Using Dependency Models to Manage Complex Software Architecture*, in "Proceedings of OOPSLA'05", 2005, pp. 167–176

[70] G. SANTOS, N. ANQUETIL, A. ETIEN, S. DUCASSE, M. TULIO VALENTE. *System Specific, Source Code Transformations*, in "31st IEEE International Conference on Software Maintenance and Evolution (ICSME)", Bremen, Germany, September 2015, 10 p. , https://hal.inria.fr/hal-01185637

[71] G. SANTOS, A. ETIEN, N. ANQUETIL, S. DUCASSE, M. TULIO VALENTE. *Recording and Replaying System Specific, Source Code Transformations*, in "15th IEEE International Working Conference on Source Code Analysis and Manipulation (SCAM)", Bremen, Germany, September 2015, 10 p. , https://hal.inria.fr/hal-01185639

[72] N. SCHÄRLI, A. P. BLACK, S. DUCASSE. *Object-oriented Encapsulation for Dynamically Typed Languages*, in "Proceedings of 18th International Conference on Object-Oriented Programming Systems, Languages and Applications (OOPSLA'04)", October 2004, pp. 130–149 [*DOI :* 10.1145/1028976.1028988], http://scg.unibe.ch/archive/papers/Scha04bOOEncapsulation.pdf

[73] N. SCHÄRLI, S. DUCASSE, O. NIERSTRASZ, A. P. BLACK. *Traits: Composable Units of Behavior*, in "Proceedings of European Conference on Object-Oriented Programming (ECOOP'03)", LNCS, Springer Verlag, July 2003, vol. 2743, pp. 248–274 [*DOI :* 10.1007/B11832], http://scg.unibe.ch/archive/papers/Scha03aTraits.pdf

[74] C. SMITH, S. DROSSOPOULOU. *Chai: Typed Traits in Java*, in "Proceedings ECOOP 2005", 2005

[75] G. SNELTING, F. TIP. *Reengineering Class Hierarchies using Concept Analysis*, in "ACM Trans. Programming Languages and Systems", 1998

[76] K. J. SULLIVAN, W. G. GRISWOLD, Y. CAI, B. HALLEN. *The Structure and Value of Modularity in SOftware Design*, in "ESEC/FSE 2001", 2001

[77] D. VAINSENCHER. *MudPie: layers in the ball of mud*, in "Computer Languages, Systems & Structures", 2004, vol. 30, n° 1-2, pp. 5–19

[78] N. WILDE, R. HUITT. *Maintenance Support for Object-Oriented Programs*, in "IEEE Transactions on Software Engineering", December 1992, vol. SE-18, n° 12, pp. 1038–1044