Actes des journées du GDR GPL 2022

Groupement de Recherche « Génie de la Programmation et du Logiciel »

Vannes, 7-10 juin 2022

Comité d’organisation

- Mireille Blay-Fornarino
- Isabelle Borne
- Catherine Dubois
- Nikolai Kosmatov
- Nicolas Magaud
- Pascal Poizat
- Responsables des groupes de travail
- TBC

Editeurs

- Mireille Blay-Fornarino
- Catherine Dubois
- Nikolai Kosmatov

Préface

C’est avec grand plaisir que nous vous accueillons pour les treizièmes journées nationales du GDR « Génie de la Programmation et du Logiciel » (GPL) qui se déroulent à Vannes du 7 au 10 juin 2022.

TODO

….

Mireille Blay-Fornarino et Catherine Dubois

Co-directrices du GDR « Génie de la Programmation et du Logiciel »

Table des matières

Invité

The upcoming wall of software complexity in computational sciences

Vincent Reverdy *† [1]

[1] Laboratoire d'Annecy de Physique des Particules – Institut National de Physique Nucléaire et de Physique des Particules du CNRS, Université Savoie Mont Blanc, Centre National de la Recherche Scientifique : UMR5814 – France

Over the last decades, computational scientists have been building high performance scientific codes to make the most of large scale computing resources. These codes form today a rich computing stack exploited to simulate and analyze phenomena in a wide variety of scientific domains. But while research communities were putting a tremendous amount of effort to add numerical methods and solvers on one side, and parallel strategies to exploit heterogeneous architectures on the other side, codes were slowly becoming more complex. In this talk, I will argue that in a growing number of computational sciences, software complexity has become a far more limiting factor than pure computing power. I will discuss the underlying mechanisms that are leading to a wall of complexity that will put heavy constraints on scientific progress. I will review the related challenges from both the computational side and the computer science side, and suggest some interdisciplinary approaches to overcome this upcoming wall. I will discuss in particular how work on software architecture and programming languages can be key to solve this problem. Throughout this talk, numerical cosmology will be taken as a canonical example since the field is facing complexity challenges that no supercomputer can solve. Before concluding on how computer scientists and researchers in application domains can to collaborate to get around the wall of complexity, I will present a few lessons that have been learned over the years in the design of Embedded Domain Specific Languages to try to reduce software complexity in astrophysics codes.

*Intervenant

†Auteur correspondant: vincent.reverdy@lapp.in2p3.fr

Du génie logiciel pour le domaine militaire

Jérémy Buisson * [1,2]

[1] Centre de recherche des écoles de Saint-Cyr Coëtquidan [Guer] (CREC) – Ecoles de Saint-Cyr Coëtquidan [Guer] – Écoles de Saint-Cyr Coëtquidan - 56381 GUER Cedex, France
[2] Institut de Recherche en Informatique et Systèmes Aléatoires (IRISA) – Universite de Rennes 1, Université de Rennes, Institut National des Sciences Appliquées - Rennes, Institut National des Sciences Appliquées, Université de Bretagne Sud, École normale supérieure - Rennes, Institut National de Recherche en Informatique et en Automatique, CentraleSupélec : UMR6074, Centre National de la Recherche Scientifique, IMT Atlantique Bretagne-Pays de la Loire, Institut Mines-Télécom [Paris] – Avenue du général LeclercCampus de Beaulieu 35042 RENNES CEDEX, France

Après une rapide présentation de l'académie militaire de Saint-Cyr Coëtquidan, je présenterai un panorama des travaux de recherches qui y sont conduits, et faisant la jonction entre le génie logiciel et le domaine militaire. Nous commencerons par considérer l'opération militaire elle-même. Le processus d'élaboration d'un ordre d'opération peut être mis en parallèle des processus d'ingénierie. On verra donc comment nous avons réutilisé les techniques de modélisation (à la UML) comme support de ce processus, et éventuellement en dériver certains artéfacts du système d'information opérationnel. Nous verrons ensuite des éléments d'ingénierie des programmes d'armement, avec une présentation succincte de NAF (un cadre d'architecture système de systèmes) et comment on peut élaborer des reconfigurations dans ce contexte. Dans le domaine militaire, les systèmes sont nécessairement sociotechniques, avec la co-construction des scénarios opérationnels et des équipements. Je présenterai donc des travaux sur l'évaluation de la vulnérabilité humaine dans ces systèmes intégrés au cyberespace. Enfin, j'aborderai des travaux plus récents sur un cadre conceptuel pour les fake news, une des manières dont nous prenons en compte la problématique de l'influence dans les travaux de l'équipe.

*Intervenant

Atelier Dissémination

Dissémination en science du logiciel

Mathieu Acher * [1]

[1] Diversity-centric Software Engineering – Inria Rennes – Bretagne Atlantique, LANGAGE ET GÉNIE LOGICIEL – France

Il ne fait guère de doute que le logiciel mange le monde et la science, et donc que la science du logiciel sera centrale pour les années à venir.
Il faut cependant reconnaitre que ce constat n'est pas forcément connu ou partagé par le grand public, les politiciens, les instituts de recherche, ou même les scientifiques d'autres disciplines.

Comment expliquer, disséminer, ou vulgariser notre travail de recherche autour de la programmation et du logiciel ?

C'est une question "ouverte", difficile et importante.

L'objectif de cet atelier est de recenser les initiatives existantes, d'identifier ou de faire émerger des idées ou approches autour de la dissémination et du GDRGPL.

Dans un premier temps, nous parlerons de dissémination à l'IUF, de variabilité logicielle profonde, de sciences, de reproductibilité en science.

Nous nous appuierons ensuite sur les résultats du sondage https://framaforms.org/initiatives-de-mediation-en-science-du-logiciel-1653468553 qui seront analysés et commentés par les participants.

Nous effectuerons ensuite 4 "ateliers" très interactifs autour des thèmes suivants :
* Quelles sont nos audiences ?
* Audience : le cas des développeurs logiciels

* De la beauté du code

* Possibles points d'actions

*Intervenant

Atelier Practical Debugging

Practical Debugging: a hands on tutorial with Pharo

Maximilian Ignacio Willembrinck Santander * [1], Steven Costiou *

[2]

[1] CRIStAL – Research Centre in Computer Science, Signal and Automatic Control of Lille (CRIStAL UMR 9189) – France
[2] CRIStAL – Univ. Lille, Inria, CNRS, Centrale Lille, UMR 9189 CRIStAL, F-59000 Lille, France – France

Nous présenterons des techniques et outils de debugging classiques et avancées, quand les utiliser, et comment les mettre en pratique de manière systématique.

Dans une première partie, nous présenterons les outils de base communs à tout type de debugger, dans quels cas de figure et comment les utiliser. Dans une second partie, nous présenterons des techniques de debugging avancées, et comment les utiliser pour construire vos propres outils lorsque les debuggers de base ne sont plus adaptés.

Chaque technique sera illustrée par une démonstration et mise en pratique via des exercices, pour lesquels nous utiliserons le langage Pharo.

Programme de l'atelier :

Part 1 - Basic debugging. Expected duration: 40 mins

1. Basic debuggers
2. Halting & Breakpoints
3. (Creating your own) Logging Utility

Part 2 - Advanced debugging. Expected duration: 40 mins

1. Instrumentation with Reflection techniques
2. Scripting your debugger
3. Demo: time-traveling debugging using Seeker, a time-traveling debugger built in Pharo

*Intervenant

Atelier Préparation au Concours CNRS

Atelier de préparation aux concours CNRS

Catherine Dubois * [1]

[1] Ecole Nationale Supérieure dÍnformatique pour lÍndustrie et lÉntreprise – ENSIIE, Samovar – France

Cet atelier est à destiniation des jeunes chercheurs. L'atelier sera animé par Catherine Dubois et se déroulera en 2 parties :
- présentation rapide du fonctionnement du CoNRS et les jurys d'admissibilité (15min+5min de questions),
- table ronde (50min + réponses aux questions de la salle) : y participeront Thomas DEGUEULE, Jen-Louis GIAVITTO, Djamel Eddine KHELLADI et Vincent REVERDY.

*Intervenant

Session GT HIFI

Business Processes Meet Spatial Concerns: the sBPMN Verification Framework

Pascal Poizat * [1,2]

[1] LIP6 – Sorbonne Universite, Centre National de la Recherche Scientifique : UMR7606 – France
[2] Université Paris Lumières – Université Paris Nanterre – France

BPMN is the standard for business process modeling. It includes a rich set of constructs for control-flow, inter-process communication, and time-related concerns. However, spatial concerns are left apart while being essential to several application domains. We propose a comprehensive extension of BPMN to deal with this. Our proposal includes an integrated notation, a first-order logic semantics of the extension, and tool-supported verification means through the implementation of the semantics in TLA . Our tool support and our model database are open source and freely available online.

*Intervenant

Philosophers may Dine - Definitively!

Safouan Taha * [1]

[1] Laboratoire Méthodes Formelles – Institut National de Recherche en Informatique et en Automatique, CentraleSupélec, Université Paris-Saclay, Centre National de la Recherche Scientifique : UMR9021, Ecole Normale Supérieure Paris-Saclay – France

The theory of Communicating Sequential Processes (CSP) going back to Hoare is still today one of the reference theories for concurrent specification and computing. In 1997, a first formalization in Isabelle of the denotational semantics of the Failure/Divergence Model of CSP was undertaken; in particular, this model can cope with infinite alphabets, in contrast to model-checking approaches limited to finite ones. In this paper, we extend this theory to a significant degree by taking advantage of more powerful automation of modern Isabelle version, which came even closer to recent developments in the semantic foundation of CSP. Better definitions allow to clarify a number of obscure points in the classical literature, for example concerning the relationship between deadlock- and livelock-freeness.

As a result, we have a modern environment for formal proofs of concurrent systems. We demonstrate a number of verification-techniques for classical, generalized examples like the Dijkstra's Dining Philosopher Problem of an arbitrary size.

*Intervenant

Formal Methods in Practice: Model Checking in the Railway Industry

Nicolas Aucouturier * [1]

[1] Prover Technology – Prover Technology – France

The railway industry is a long term user of formal methods, especially in France. For more than 30 years, both infrastructure managers and industrial suppliers are intensively using formal tools to increase confidence in the critical parts of railway systems.This is strongly encouraged by the certification authorities, since the standard in the area highly recommends such application. Most famous success stories in this context involve B Method (Siemens/RATP for Line 14 and others), SCADE with corresponding tools (eg Thales for Line 13), and HLL the current standard language required in tenders (eg for ARGOS SNCF). Prover provides formal methods solutions for 30 years and in particular a tool suite for HLL including a model-checker based on SAT solvers. In this talk, we will present the typical activities performed on a daily basis, and the systems along with their properties that are currently analyzed. We will also present some trends and further goals.

*Intervenant

Session GT GLIA

What's new about Wise Object

Flavien Vernier * [1]

[1] Laboratoire dÍnformatique, Systèmes, Traitement de lÍnformation et de la Connaissance (LISTIC) – Université Savoie Mont Blanc : EA3703 – BP 80439 74944 ANNECY LE VIEUX Cedex, France

Lors de cette présentation, les évolutions conceptuelles et architecturales du framework des Objets Sages seront abordées, notamment celles de découplage métier, intelligence et sémantique. La présentation se focalisera ensuite sur les travaux de thèse en cours et à venir, c'est-à-dire i) les premières étapes pour la recherche de liens sémantiques entre certaines connaissances, ii) les problématiques liées à l'application des WO dans la détection de fraudes bancaires et iii) l'implémentation de chaînes de traitement scientifique sages.

*Intervenant

Adversarial retraining pour les systèmes configurables

Paul Temple * [1]

[1] PReCISE research center, University of Namur – Belgique

Les systèmes deviennent de plus en plus complexes et peuvent être configurés afin de répondre au mieux aux attentes des utilisateurs tout en capitalisant sur la réutilisabilité du code. Le problème étant que les espaces de configuration deviennent si grand qu'il est impossible d'explorer, générer, tester toutes les configurations possibles de ces systèmes. L'utilisation de modèles de Machine Learning (ML) permet alors d'élager cet espace suivant un but donné (e.g., garder les configurations qui n'utilisent pas plus d'un certain seuil en espace mémoire). Le but étant de réduire drastiquement mais également de manière précise le nombre de configurations à évaluer. En parallèle, depuis le début des années 2010, l'évolution extrêmement rapide du ML à poser de nombreuses questions y compris concernant le fonctionnement de ces modèles. Le domaine de l'adversarial machine learning a alors émergé et cherche à comprendre comment des personnes malveillantes peuvent exploiter les faiblesses des modèles de ML et comment s'en défendre. Dans sa réalisation, l'adversarial machine learning forge de nouvelles données ayant un impact maximal sur les modèles de ML (e.g., maximiser les chances d'être mal classée). Nous proposons d'utiliser ces techniques sur des systèmes configurables (et les modèles de ML associées) afin de générer automatiquement de nouvelles configurations du système. Cette génération permet entre autre : de rendre le modèle de ML, utilisé pour filtrer les configurations, plus précis; d'explorer davantage l'espace de configuration du système. Ce travail a été publié à EMSE 2021 (DOI: https://doi.org/10.1007/s10664-020-09915-7).

*Intervenant

Extraction de la variabilité depuis les schémas de données des entrées/sorties de simulateurs de systèmes d'aide à la décision dans le domaine agricole

Thomas Georges * [1]

[1] Laboratoire dÍnformatique de Robotique et de Microélectronique de Montpellier – Université de Montpellier : UMR5506, Centre National de la Recherche Scientifique : UMR5506 – France

Le contexte de ces travaux est le développement de logiciels pour l'aide à la décision en agronomie par notre collaborateur industriel ITK. Ces logiciels aident les agriculteurs à comprendre et anticiper l'état des cultures grâce à diverses prédictions de simulateurs. Chaque simulateur possède ses propres paramètres à fournir en entrée et reçus en sortie de simulation, avec par exemple le simulateur de production demandant des données différentes du simulateur de la maladie pour la vigne tout en partageant aussi des données communes telles que la météo. Le projet sur lequel nous travaillons a pour but de construire une ligne de produits logiciels qui permettra : i) une dérivation simple de nouveaux produits (par les équipes IT) à partir de nouveaux simulateurs (faits par les équipes d'agronomes), et ii) de simplifier la maintenance de la base de code existante chez ITK. Le processus d'extraction peut être laborieux et prendre beaucoup de temps. C'est pourquoi nous étudions les moyens d'automatiser ce processus en se concentrant sur les schémas de données décrivant les paramètres à fournir en entrée et reçus en sortie des simulations. Nous avons fait l'hypothèse que l'analyse formelle de concept (FCA) est un outil utile pour l'extraction de la variabilité des logiciels, avec par exemple la mise en évidence des points communs et des spécificités de ces logiciels nous permettant d'assister les équipes d'IT et d'agronomes dans la construction de nouveaux logiciels et simulateurs. Cet exposé présente nos travaux sur ce processus permettant d'extraire la variabilité. C'est un processus basé sur une série de pré-traitements permettant l'utilisation d'outils d'analyse formelle de concepts. Ces outils sont utilisés pour construire un AOC-Poset, une structure conceptuelle dérivée d'un treillis de concept dans laquelle nous avons identifié les caractéristiques communes et variables des simulateurs à partir de leurs schémas de données. Nous avons implémenté ce processus, l'avons expérimenté sur une collection de 6 simulateurs et avons obtenu des résultats prometteurs pour la future construction de la ligne de produit.

*Intervenant

Session GT CLAP

Macle : un langage dédié à l'accélération de programmes OCaml sur FPGA

Loic Sylvestre * [1], Emmanuel Chailloux [1], Jocelyn Serot [2]

[1] LIP6 – Sorbonne Universite, Centre National de la Recherche Scientifique : UMR7606 – France
[2] Institut Pascal – CNRS : UMR6602, Université Blaise Pascal - Clermont-Ferrand II – 24 avenue des Landais 63171 Aubiere Cedex, France

O2B (OCaml On Board) est un portage de l'implémentation OMicroB de la machine virtuelle OCaml ciblant un processeur softcore réalisé sur du matériel configurable de type FPGA (Field-programmable gate array). Cela rend possible, en OCaml, l'appel de circuits (vus comme des fonctions externes) directement implantés sur le FPGA, notamment à des fins d'accélération matérielle. Macle (ML accelerator) est un langage dédié à la programmation de haut-niveau de tels circuits de calcul interopérants avec OCaml.

*Intervenant

Optimisations dans le compilateur formellement vérifié CompCert

David Monniaux * [1]

[1] VERIMAG – Centre National de la Recherche Scientifique : UMR5104, Université Grenoble Alpes, Institut polytechnique de Grenoble (Grenoble INP) – France

CompCert est un compilateur de C vers de multiples architectures cibles, avec l'originalité d'une preuve de préservation de la sémantique des programmes : si un programme C a une exécution bien définie (succession d'appels de fonctions externes, d'accès aux variables volatiles...), alors le programme compilé suit la même exécution. Cette preuve s'obtient par composition de relations de simulation (pas à pas ou plus complexes) entre les différentes représentations intermédiaires, reliées par les passes de transformation et d'optimisation. Elle est vérifiée à l'aide de l'assistant Coq.
La version "officielle" de CompCert (diffusée sur github et achetable auprès de la société Absint) est modérément optimisante. À Verimag, nous avons rajouté diverses optimisations permettant des gains de performance notables, notamment des ordonnancements d'instructions adaptés à divers processeurs, dont des processeurs VLIW et des éliminations de code redondant.
La difficulté est qu'il ne suffit pas d'implanter des méthodes existantes telles que celles présentes dans gcc et LLVM, mais qu'il faut dégager une approche de preuve qui passe à l'échelle et soit facile à maintenir par la suite. Notre preuve de correction de l'ordonnancement est basé sur un vérificateur prouvé, fonctionnant par exécution symbolique.

*Intervenant

Modern Compiler Technology to Optimize Code from Ionic Models

Tiago Trevisan Jost * [1], Arun Thangamani [1], Vincent Loechner [2], Stéphane Genaud [2], Bérenger Bramas [1]

[1] Unistra et Inria NGE – Université de Strasbourg (UNISTRA) – France
[2] Université de Strasbourg, UFR de mathématique et informatique (unistra) – www.unistra.fr – France

The MLIR compiler framework is a novel compiler infrastructure that eases the process of developing new interacting compiler transformations and intermediate representations, built as part of the LLVM framework. In this talk, I will present our experience using MLIR within the MICROCARD (https://microcard.eu/), a European project aimed at building and simulating cardiac electrophysiology using whole-heart models. The project builds on the existing open-source openCARP project (opencarp.org), a cardiac electrophysiology simulator for in-silico experiments. OpenCARP includes a solver and the ionic model component describing ionic transmembrane currents, as ordinary differential equations (ODEs). They are provided using a DSL (domain-specific language) for ODEs named easyML. This talk will cover how we have used the MLIR infrastructure, its dialects, and transformations to drive forward the study of ionic models, and accelerate the execution of multi-cell systems.

*Intervenant

Session GT IE & AFSEC

Couplage des approches MBSE et MDAO pour le dimensionnement d'une batterie de drone

Ombeline Aïello * [1,2]

[1] Institut Supérieur de lÁéronautique et de lÉspace – DGA – France
[2] DTIS, ONERA [Toulouse] – ONERA – France

Les drones sont de plus en plus utilisés pour suppléer les humains lors de missions de sauvetage et de surveillance. Leur conception s'avère être un défi, en particulier lorsqu'il faut dimensionner les batteries du drone en fonction de l'autonomie attendue pour réaliser une mission donnée. C'est pourquoi, des solutions sont à rechercher pour concevoir un drone de manière rigoureuse tout en spécifiant ses principales caractéristiques. Cet article présente un travail en cours visant à combler le fossé entre deux disciplines d'ingénierie qui ont jusqu'à présent été développées séparément : l'ingénierie des systèmes basée sur les modèles (MBSE, Model-Based System Engineering en anglais) et, l'analyse et l'optimisation multidisciplinaire (MDAO, Multidisciplinary Design Analysis and Optimization en anglais). Le couplage du MBSE et de la MDAO est abordé en termes de langage, d'outils et de méthodes.

*Intervenant

Concevoir des patrons de justification pour la certification

Thomas Polacsek * [1]

[1] ONERA/DTIS, Toulouse – ONERA – France

L'une des conditions préalables à l'introduction sur le marché d'un système critique est sa certification par les autorités de régulation. Pour ce faire, l'organisation "candidate" doit démontrer la conformité de son produit aux normes et standards du domaine. La grande complexité de ce processus a conduit l'organisation candidate à s'appuyer sur des formes structurées d'argumentation que cela soit dans le domaine médical, nucléaire ou aéronautique. Dans cet article, nous proposons une méthode générique qui guide l'organisation candidate dans la spécification de patrons d'argumentation structurée pour une norme. Contrairement aux travaux existants qui se concentrent sur un seul contexte, notre objectif est de fournir une approche à la fois générique et indépendante du domaine. Afin d'illustrer cette nouvelle approche, nous présentons les résultats de son application sur un cas d'étude réel.

*Intervenant

Spécification formelle de systèmes cyber-physiques et Ingénierie système assistée par la simulation

Thuy Nguyen * [1]

[1] EDF – EDF – France

La méthode BASAALT (Behaviour Analysis and Simulation All Along systems Life Time) a été concçue pour aider l'ingénierie des systèmes cyber-physiques et socio-techniques complexes, ainsi que des grands systèmes de systèmes . Elle s'intéresse aux comportements et autres phénomènes dynamiques comme la sûreté de fonctionnement et les coûts d'exploitation, et part du principe que pour pour ce type de systèmes, un support outillé important est nécessaire : comme pour les logiciels, la seule inspection manuelle des modèles ne permet pas de révéler la plupart de leurs défauts, et le test / simulation ou la vérification formelle sont des aides précieuses. Pour cela, les exigences et les solutions doivent être modélisées formellement, dans un langage compréhensible par les personnes impliquées.

FORM-L (FOrmal Requirements Modelling Language) est le langage de modélisation de BASAALT. C'est un langage de contrainte non déterministe qui peut être utilisé dès les phases les plus en amont du cycle de vie et qui permet d'éviter la surspécification inhérente aux langages déterministes.

L'ingénierie des exigences est un aspect important de BASAALT, car l'expérience montre que même pour les systèmes les plus critiques, les exigences sont souvent inadéquates avec parfois des conséquences inacceptables, voire catastrophiques. Il est donc important de ne pas se contenter de la forme et de s'intéresser de près à leur fond et à leur sémantique. Par ailleurs, pour BASAALT, la spécification et la justification des hypothèses sont aussi importantes que la spécification des exigences.

*Intervenant

Automatic Support for Requirements Validation

Rabéa Ameur-Boulifa * [1]

[1] Télécom Paris – Institut Polytechnique de Paris – 19 Place Marguerite Perey 91120 Palaiseau, France

The automotive industry is currently going through rapid changes from a mechanical industry to one driven by innovation in electronics and embedded software. This significant change creates also significant challenges to the industry. One of the most important is the ability to create safe vehicles, emphasizing the importance of safety by design.

In this context, we propose a framework based on a correction-by-design approach the industry-wide development of reliable systems. The tool aims to integrate formal analysis and verification of requirements at the earliest stages of the development life cycle. This work introduces a systematic process for the unambiguous specification of systems and the guided derivation of models that are evidences that the requirement specifications are realizable. This rigorous design is carried out by incremental model building using model-checking tool.

*Intervenant

Session GT IDM

Self-Adaptable Languages

Gwendal Jouneaux * [1], Olivier Barais [1], Benoit Combemale [1], Gunter Mussbacher [2,3]

[1] Université de Rennes 1 – Université de Rennes I – France
[2] Inria – L'Institut National de Recherche en Informatique et e n Automatique (INRIA) – France
[3] McGill University – Canada

Over recent years, self-adaptation has become a concern for many software systems that have to operate in complex and changing environments. At the core of self-adaptation, there is a feedback loop and 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, this paper introduces the concept of Self-Adaptable Language (SAL) to abstract the feedback loops at both system and language levels. We propose L-MODA (Language, Models, and Data) as a conceptual reference framework that characterizes the possible feedback loops abstracted into a SAL. To demonstrate SALs, we present emerging results on the abstraction of the system feedback loop into the language semantics. We report on the concept of Self-Adaptable Virtual Machines as an example of semantic adaptation in a language interpreter and present a roadmap for SALs.

*Intervenant

Les notions clés de l'assistance à la modélisation logicielle

Maxime Savary-Leblanc * [1,2]

[1] Centre de Recherche en Informatique, Signal et Automatique de Lille - UMR 9189 – Université de Lille : UMR9189, Centrale Lille : UMR9189, Centre National de la Recherche Scientifique : UMR9189 – France

[2] Inria – L'Institut National de Recherche en Informatique et e n Automatique (INRIA) – France

Nous sommes probablement tous d'accord pour dire que les outils de modélisation ne sont pas à la hauteur de nos attentes. Interfaces désuètes, manque d'utilisabilité, nombre de fonctionnalité trop important, ou courbe d'apprentissage trop forte sont quelques facteurs qui sont notamment responsables de cette désaffection. En parallèle de ces problèmes d'interface, ces outils ne permettent seulement aux utilisateurs que de digitaliser leur modèle, sans proposer de fonctionnalité les aidant à concevoir - c'est à dire à affiner la représentation mentale de la solution - ou les aidant à faire les bons choix d'abstraction. J'ai dédié ces quatre dernières années, incluant ma thèse, à comprendre les besoins d'assistance qu'ont les concepteurs lorsqu'ils modélisent, et à notamment cerner ce que pourrait être un assistant logiciel à la modélisation logicielle. Dans cet exposé, je présente un prototype d'assistant logiciel, et je présente les quatre notions clés derrière le concept d'assistance à la modélisation : confiance, créativité, recommandations et automatisation.
English version: Assisting software modeling: the key concepts behind the scene

We probably all agree that modeling tools are not up to our expectations. Outdated interfaces, lack of usability, too many features, or a steep learning curve are some of the factors that are responsible for this disaffection. In parallel to these interface problems, these tools only allow users to digitize their model, without offering any functionality to help them design - i.e. to refine the mental representation of the solution - or to help them make the right abstraction choices. I have dedicated the last four years, including my thesis, to understand the needs of designers for assistance when they model, and in particular to identify what could be a software assistant for software modeling. In this talk, I present a prototype of a software assistant, and I introduce the four key notions behind the concept of modeling assistance: trust, creativity, recommendations and automation.

*Intervenant

Moose : une plate-forme IDM pour l'exploration et la visualisation de programmes et de modèles

Vincent Aranega * [1], Clotilde Toullec [2]

[1] CRIStAL – Univ. Lille, CNRS, Inria, Centrale Lille, UMR 9189 CRIStAL, F-59000 Lille, France – France
[2] INRIA-Lille – L'Institut National de Recherche en Informatique et e n Automatique (INRIA) – France

La plate-forme Moose regroupe un ensemble d'outils pour l'exploration rapide de modèles et leurs visualisations sous différentes formes. Entièrement codée en Pharo, un dialecte Smalltalk, elle permet de bénéficier des avantages du live coding, de la flexibilité des langages dynamiques et d'un framework extensible pour la définition de nouveaux outils. Cette présentation se concentre sur une introduction rapide des briques logicielles et des concepts sur lesquels reposent Moose, la définition d'un métamodèle dans la plate-forme, ainsi que la démonstration de l'exploration d'un modèle issue d'un mining de dépôts github.

*Intervenant

Session GT VL

Empirical Assessment of Multimorphic Testing

Paul Temple * [1]

[1] University of Namur – Belgique

The performance of software systems such as speed, memory usage, correct identification rate, tends to be an evermore important concern, often nowadays on par with functional correctness for critical systems. Systematically testing these performance concerns is however extremely difficult, in particular because there exists no theory underpinning the evaluation of a performance test suite, i.e., to tell the software developer whether such a test suite is "good enough" or even whether a test suite is better than another one. 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. 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 3 empirical studies carried out in different fields: object tracking in videos, object recognition in images, and code generators.

*Intervenant

Rotten green tests in Java, Pharo and Python

Vincent Aranega * [1]

[1] CRIStAL – Univ. Lille, CNRS, Centrale Lille Institut – France

Rotten Green Tests are tests that pass, but not because the assertions they contain are true: a rotten test passes because some or all of its assertions are not actually executed. The presence of a rotten green test is a test smell, and a bad one, because the existence of a test gives us false confidence that the code under test is valid, when in fact that code may not have been tested at all. This article reports on an empirical evaluation of the tests in a corpus of projects found in the wild. We selected approximately one hundred mature projects written in each of Java, Pharo, and Python. We looked for rotten green tests in each project, taking into account test helper methods, inherited helpers, and trait composition. Previous work has shown the presence of rotten green tests in Pharo projects; the results reported here show that they are also present in Java and Python projects, and that they fall into similar categories. Furthermore, we found code bugs that were hidden by rotten tests in Pharo and Python. We also discuss two test smells -*missed fail* and *missed skip* -that arise from the misuse of testing frameworks, and which we observed in tests written in all three languages.

*Intervenant

Breaking bad? Semantic versioning and impact of breaking changes in Maven Central

Thomas Degueule * [1]

[1] Laboratoire Bordelais de Recherche en Informatique – Université de Bordeaux, Centre National de la Recherche Scientifique : UMR5800 / URA1304, École Nationale Supérieure d'Électronique, Informatique et Radiocommunications de Bordeaux (ENSEIRB) – France

Just like any software, libraries evolve to incorporate new features, bug fixes, security patches, and refactorings. However, when a library evolves, it may break the contract previously established with its clients by introducing Breaking Changes (BCs) in its API. These changes might trigger compile-time, link-time, or run-time errors in client code. As a result, clients may hesitate to upgrade their dependencies, raising security concerns and making future upgrades even more difficult. Understanding how libraries evolve helps client developers to know which changes to expect and where to expect them, and library developers to understand how they might impact their clients. In the most extensive study to date, Raemaekers et al. investigate to what extent developers of Java libraries hosted on the Maven Central Repository (MCR) follow semantic versioning conventions to signal the introduction of BCs and how these changes impact client projects. Their results suggest that BCs are widespread without regard for semantic versioning, with a significant impact on clients. In this paper, we conduct an external and differentiated replication study of their work. We identify and address some limitations of the original protocol and expand the analysis to a new corpus spanning seven more years of the MCR. We also present a novel static analysis tool for Java bytecode, Maracas, which provides us with: (i) the set of all BCs between two versions of a library, and; (ii) the set of locations in client code impacted by individual BCs. Our key findings, derived from the analysis of 119,879 library upgrades and 293,817 clients, contrast with the original study and show that 83.4% of these upgrades do comply with semantic versioning. Furthermore, we observe that the tendency to comply with semantic versioning has significantly increased over time. Finally, we find that most BCs affect code that is not used by any client, and that only 7.9% of all clients are affected by BCs. These findings should help (i) library developers to understand and anticipate the impact of their changes; (ii) library users to estimate library upgrading effort and to pick libraries that are less likely to break, and; (iii) researchers to better understand the dynamics of library-client co-evolution in Java.

*Intervenant

Session GT YODA

Models and Verification for Composition and Reconfiguration of Web of Things Applications

Gwen Salaün * [1]

[1] Univ. Grenoble Alpes, CNRS, Grenoble INP, Inria, LIG – CNRS : UMR5217, L'Institut National de Recherche en Informatique et e n Automatique (INRIA), Institut polytechnique de Grenoble (Grenoble INP), Université Grenoble Alpes, LIG – France

The Internet of Things (IoT) applications are built by interconnecting everyday objects over a network. These objects or devices sense the environment around them, and their network capabilities allow them to communicate with other objects to perform utilitarian tasks. One of the popular ways to build IoT applications in the consumer domain is by combining different objects using Event-Condition-Action (ECA) rules. These rules are typically in the form of IF something-happens THEN do-something. The Web of Things (WoT) are a set of standards and principles that integrate architectural styles and capabilities of web to the IoT. Even though WoT architecture coupled with ECA rules simplifies the building of IoT applications to a large extent, there are still challenges in making end-users develop advanced applications in a simple yet correct fashion due to dynamic, reactive and heterogeneous nature of IoT systems. The broad objective of this work is to leverage formal methods to provide end-users of IoT applications certain level of guarantee at design time that the designed application will behave as intended upon deployment. In this context, we propose a formal development framework based on the WoT. The objects are described using a behavioural model derived from the Thing Description specification of WoT. Then, the applications are designed not only by specifying individual ECA rules, but also by composing these rules using a composition language. The language enables users to build more expressive automation scenarios. The description of the objects and their composition are encoded in a formal specification from which the complete behaviour of the application is identified. In order to guarantee correct design of the application, this work proposes a set of generic and application-specific properties that can be validated on the complete behaviour before deployment. Further, the deployed applications may be reconfigured during their application lifecycle. The work supports reconfiguration by specifying reconfiguration properties that allow one to qualitatively compare the behaviour of the new configuration with the original configuration. The implementation of all the proposals is achieved by extending the Mozilla WebThings platform. A new set of user interfaces have been built to support the composition of rules and reconfiguration. A model transformation component which transforms WoT models to formal models and an integration with formal verification toolbox are implemented to enable automation. Finally, a deployment engine is built by extending WebThings APIs. It directs the deployment of applications and reconfigurations respecting their composition semantics.

*Intervenant

Reusability of Autonomic Controllers in High Performance Computing

Quentin Guilloteau * [1]

[1] Universiteé Grenoble Alpes, Inria, CNRS, Grenoble INP, LIG – Université Grenoble Alpes, L'Institut National de Recherche en Informatique et e n Automatique (INRIA), CNRS : UMR5217, Institut polytechnique de Grenoble (Grenoble INP), LIG – France

High Performance Computing systems are subject to dynamical variations occurring in e.g., the execution of the jobs, the quantity of IO, the network consumption, etc. Such systems thus need Autonomic management with online regulation. In our work, we consider High Performance Computing (HPC) systems, and more particularly CiGri, a system harvesting idle resources in a computing grid. It submits jobs from Bag-of-Tasks applications with the lowest priority to the clusters in order to maximize their use. This harvesting introduces perturbations for the premium users of the cluster, e.g., degraded performances of the fileserver. Therefore, the submission must be regulated in a feedback loop with a Control Theory approach. The design of such a controller can involve considerable competence and work, and it is highly desirable that it can be as reusable as possible, adaptable to different contexts where it is instantiated. In this paper we propose to study the relationship between this reusability and the techniques of adaptive control, by describing an approach involving classical PID control and Model-Free Control (MFC), and performing experimental validation and comparison in the use-case of CiGri, evaluating several criteria and in particular their reusability and adaptivity to variations.

*Intervenant

Session GT Debugging

SciHook: A Language-Agnostic Python-Based Instrumentation Library for Scientific Computing

Benoit Combemale * [1]

[1] Diverse – Univ Rennes, CNRS, Inria, IRISA - UMR 6074 – France

*

*Intervenant

Debuggable test cases for domain-specific models

Gerson Sunyé * [1]

[1] NaoMod - Nantes Software Modeling Group – Laboratoire des Sciences du Numérique de Nantes – France

*

*Intervenant

Bug Stories

Steven Costiou * [1]

[1] CRIStAL – Univ. Lille, Inria, CNRS, Centrale Lille, UMR 9189 CRIStAL, F-59000 Lille, France – France

Cette petite présentation introduira la session du GT Debugging, et pour s'échauffer nous raconterons quelques histoires de bugs terrifiantes...

*Intervenant

Deboguage du compilateur C vérifié CompCert

David Monniaux * [1]

[1] VERIMAG – Centre National de la Recherche Scientifique : UMR5104, Université Grenoble Alpes, Institut polytechnique de Grenoble (Grenoble INP) – France

CompCert est un compilateur C formellement vérifié, c'est-à-dire qu'il y a une preuve mathématique que le code produit correspond au code source, au sens qu'il produit les mêmes exécutions. Toutefois, il contient des bugs, qui parfois conduisent à générer du code incorrect. Cela peut paraître paradoxal ; nous expliquerons comment cela est possible, et ce que nous avons fait pour rechercher des bugs (ou parfois en trouver un peu par hasard), et ce qu'il faudrait faire pour aller plus loin.

*Intervenant

Session AFADL & MTV2

Test aléatoire et énumératif pour OCaml et Why3

Alain Giorgetti [1], Jérome Ricciardi * [2,3], Clotilde Erard [4]

[1] Franche-Comté Électronique Mécanique, Thermique et Optique - Sciences et Technologies (UMR 6174) – Université de Franche-Comté, Ecole Nationale Supérieure de Mécanique et des Microtechniques, Centre National de la Recherche Scientifique : UMR6174, Université de Technologie de Belfort-Montbeliard : UMR6174 – France
[2] CEA - List – CEA-LIST – France
[3] Laboratoire Méthodes Formelles – Université Paris-Saclay – France
[4] Franche-Comté Électronique Mécanique, Thermique et Optique - Sciences et Technologies (UMR 6174) – Université de Franche-Comté, Ecole Nationale Supérieure de Mécanique et des Microtechniques, Centre National de la Recherche Scientifique : UMR6174, Université de Technologie de Belfort-Montbeliard : UMR6174 – France

Nous présentons AutoCheck, un prototype d'outil de test aléatoire et énumératif de propriétés définies dans le langage fonctionnel OCaml ou dans le langage WhyML de la plateforme de vérification déductive Why3. Une originalité est que les tests énumératifs utilisent des générateurs de données eux-mêmes écrits dans le langage WhyML, et dont la correction et la complétude sont formellement prouvées avec Why3. Une autre spécificité est que l'effort de développement est réduit en exploitant le mécanisme d'extraction de Why3 vers OCaml et un outil de test aléatoire existant pour OCaml.

*Intervenant

De l'adaptation de Caseine pour l'évaluation des tests des étudiants

Lydie Du Bousquet * [1]

[1] Laboratoire d'Informatique de Grenoble – Centre National de la Recherche Scientifique : UMR5217, Université Grenoble Alpes, Institut polytechnique de Grenoble - Grenoble Institute of Technology – France

Nous présentons un environnement et une approche pour l'enseignement du test. L'enseignement du test est un exercice délicat. On peut choisir d'enseigner la philosophie (e.g., "rechercher des erreurs"), les outils (e.g., JUnit, ...), des objectifs (e.g., correction fonctionnelle, performance, non régression, ...), des phases de tests (e.g. unitaire, intégration, ...), et/ou des méthodes de sélection de données. L'évaluation du travail des étudiants se base généralement sur l'évaluation de la qualité des tests. Pour cela, on s'appuie classiquement sur une approche d'analyse mutationnelle ou d'injection de fautes. Dans les deux cas, des programmes alternatifs au programme sous test sont construits en introduisant des fautes. Les tests sont ensuite exécutés sur ces programmes fautifs. On estime que les tests sont de qualité lorsque toutes les fautes sont découvertes. Nous avons étendu l'environnement Caseine et l'approche pour l'évaluation des tests des étudiants.

*Intervenant

Présentation des résultats du Projet ANR AAPG 2018 – PHILAE From Model-Based Testing to Cognitive Test Automation

Bruno Legeard * [1]

[1] Franche-Comté Électronique Mécanique, Thermique et Optique - Sciences et Technologies (UMR 6174) – Université de Franche-Comté, Ecole Nationale Supérieure de Mécanique et des Microtechniques, Centre National de la Recherche Scientifique : UMR6174, Université de Technologie de Belfort-Montbeliard : UMR6174 – France

Présentation des résultats du Projet ANR AAPG 2018 – PHILAE From Model-Based Testing to Cognitive Test Automation.

*Intervenant

Analyse automatisée de binaires à la recherche de vulnérabilités matérielles

Théo De Castro Pinto * [1]

[1] Laboratoire Bordelais de Recherche en Informatique – Université de Bordeaux, Centre National de la Recherche Scientifique : UMR5800 / URA1304, École Nationale Supérieure d'Électronique, Informatique et Radiocommunications de Bordeaux (ENSEIRB) – France

Les cartes à puce et autres appareils embarqués mettent les développeurs face à un problème de taille : les attaques physiques. Parmi celles-ci, certaines visent à modifier le comportement de la puce (temporairement ou non) grâce à une perturbation physique (laser, glitch ou autre). Ces dernières, qui peuvent être multiples au cours d'une seule exécution, sont difficiles à détecter et même si des contre-mesures existent, elles peuvent être insuffisantes ou disparaitre à la compilation. Pour ces raisons une analyse des binaires générés est nécessaire, afin de détecter la présence (ou non) de vulnérabilités face à ces attaques. Cette analyse est généralement effectuée à la main. Ce papier présente un outil d'aide à la recherche de vulnérabilités matérielles pour du code binaire. Il s'appuie sur l'outil Binsec. Il permet de détecter des vulnérabilités dans un binaire en fonction d'un schéma d'attaque et d'une cible à atteindre. Il s'agit de résultats préliminaires obtenus lors d'un stage de Master. Ce papier présente aussi quelques problématiques de recherche qui seront abordées au cours de la thèse.

*Intervenant

Session AFADL & LVP

Vérification de l'algorithme de calcul des ordres d'appel dans Parcoursup

Hugo Gimbert * [1], Pierre Castéran [1], Claire Mathieu [2], Gérald Point [1]

[1] LaBRI – Centre National de la Recherche Scientifique - CNRS : UMR5800 – France
[2] IRIF – Centre National de la Recherche Scientifique - CNRS : UMR8253 – France

Parcoursup est la plateforme nationale de préinscription en première année de l'enseignement supérieur en France. Ce document présente une partie des travaux de sûreté logicielle autour d'un des algorithmes de cette plateforme, appelé "calcul des ordres d'appel" : spécification de l'algorithme, vérification à l'exécution de la spécification, et preuve formelle de correction de l'algorithme. Ces travaux ont été réalisés à l'aide des outils de preuve Why3 et COQ et de techniques de vérification à l'exécution du code Java. Cela permet d'atteindre un très haut niveau de confiance dans l'algorithme de calcul des ordre d'appel de Parcoursup.

*Intervenant

Typage avancé de langages dynamiques

Mickael Laurent * [1]

[1] Université Paris Cité – CNRS – France

Avec l'essor du Web, la quantité de code écrit dans des langages dynamiques tels que JavaScript a considérablement augmenté ces dernières années. Afin d'augmenter la sûreté de tels programmes, on aimerait pouvoir les typer statiquement. En particulier, on aimerait pouvoir typer avec précision des expressions utilisant des typecases, i.e. des branchements conditionnés par le résultat d'un test de type (à l'exécution) d'une variable ou expression. Ma thèse consiste en la conception d'un tel système de type, utilisant la puissance des types ensemblistes au service de techniques de typage avancées telles que l'occurrence typing.

*Intervenant

Knit&Frog: Pattern matching compilation for custom memory representations

Thaïs Baudon * [1], Gabriel Radanne [2], Laure Gonnord [3]

[1] ENS Lyon, LIP – CNRS UMR 5668 – France
[2] Inria, LIP – CNRS UMR 5668 – France
[3] Grenoble-INP/LCIS, LIP – CNRS UMR 5668 – France

Initially present only in functional languages such as OCaml and Haskell, Algebraic Data Types have now become pervasive in mainstream languages, providing nice data abstractions and an elegant way to express functions though pattern-matching. Numerous approaches have been designed to compile rich pattern matching to cleverly designed, efficient decision trees. However, these approaches are specific to a choice of internal memory representation which must accommodate garbage-collection and polymorphism. ADTs now appear in languages more liberal in their memory representation such as Rust. Notably, Rust is now introducing more and more optimizations of the memory layout of Algebraic Data Types. As memory representation and compilation are interdependent, it raises the question of pattern matching compilation in the presence of non-regular, potentially customized, memory layouts. In this article, we present Knit&Frog, a framework to compile pattern-matching for monomorphic ADTs, parametrized by an arbitrary memory representation. We propose a novel way to describe choices of memory representation along with a validity condition under which we prove the correctness of our compilation scheme. The approach is implemented in a prototype tool ribbit.

*Intervenant

AFADL

xDSLs dirigés par les Modèles Formels : Tour d'horizon de l'outil Meeduse

Akram Idani * [1]

[1] Laboratoire d'Informatique de Grenoble (LIG/VASCO) – Laboratoire d'Infomatique de Gnoble, Equipe VASCO – UJF-Grenoble 1/Grenoble-INP/UPMF-Grenoble2/CNRS, LIG UMR 5217, F-38041, Grenoble, France, France

Meeduse est un atelier de conception formelle de langages dédiés domaine (DSLs). Il permet de travailler sur la correction d'un DSL au moyen d'outils de raisonnements automatisés. La force de l'outil provient de ProB (un animateur et un model-checker de la méthode B) et d'EMF (un environnement IDM bien établi). Cet article passe en revue deux usages utiles de Meeduse : (i) exécution et déboggage, et (ii) transformation de modèles.

*Intervenant

An Incremental Model-Based Design Methodology to Develop CPS with SysML/OCL/Reo

Perla Tannoury * [1]

[1] Franche-Comté Électronique Mécanique, Thermique et Optique - Sciences et Technologies (UMR 6174) – Université de Franche-Comté, Ecole Nationale Supérieure de Mécanique et des Microtechniques, Centre National de la Recherche Scientifique : UMR6174, Université de Technologie de Belfort-Montbeliard : UMR6174 – France

Modeling Cyber-Physical Systems (CPS) remains a challenge due to their interconnected networks of heterogeneous embedded systems that operate in a physical environment. In this paper, we introduce a new modeling approach that relies on SysML, OCL, and Reo to capture the different aspects of CPS, including requirements, architecture, and interaction protocols. The novelty of our approach relies in the combination of SysML and Reo to handle the complexity of CPS architecture and protocols, in the design step by proceeding incrementally. Furthermore, we define OCL constraints to specify rules to be respected to model consistently CPS.

*Intervenant

Spécification semi-formelle et formelle d'une application de téléréhabilitation : retour d'expérience

Farid Arfi * [1]

[1] EuroMov - Digital Health in Motion – IMT - MINES ALES, Université de Montpellier : $\mathrm{UR}_U M_I MT_1 02, IMTMinesAls - -France$

Nous nous intéressons aux étapes de recueil des besoins et de spécification d'une application de téléréhabilitation de patients atteints de maladies chroniques respiratoires. Après avoir réalisé des entretiens avec les experts impliqués dans la définition des processus de réhabilitation, un cahier des charges a été établi sous forme textuelle et sous forme de modèles UML. Nous avons constaté que la modélisation UML est un réel apport pour la maîtrise d'ouvrage mais elle doit être accompagnée d'une étape de modélisation formelle et de vérification pour analyser les processus et vérifier des propriétés. Ici nous nous concentrons sur les propriétés de vivacité et de sûreté. Nous avons choisi U PPAAL comme outil de simulation et de vérification formelle. Nous montrons sur cette étude de cas quels sont les apports de chacune des étapes de modélisation informelle et formelle pour toutes les parties prenantes du projet.

*Intervenant

Etude comparative des méthodes pour la vérification des systèmes cyber-physiques basés machine learning

Arthur Clavière * [1]

[1] Collins Aerospace – Aeospace – France

Cet article est un résumé étendu de l'article "Verification of machine learning based cyber-physical systems: a comparative study", accepté à HSCC 2022 (25th ACM International Conference on Hybrid Systems: Computation and Control).

*Intervenant

Etude de propriétés d'opacité temporisée à l'aide de vérification temporisée paramétrée

Dylan Marinho * [1]

[1] Laboratoire Lorrain de Recherche en Informatique et ses Applications – Institut National de Recherche en Informatique et en Automatique, Université de Lorraine, Centre National de la Recherche Scientifique : UMR7503 – France

Une fuite d'informations peut avoir des conséquences dramatiques sur la sécurité d'un système. Parmi ces failles, une fuite d'informations temporelle se produit lorsqu'un attaquant peut déduire des informations internes confidentielles en basant son attaque sur une observation temporisée du système. Nous modélisons ici le système par un automate temporisé, et considérons qu'un attaquant a accès (uniquement) au temps d'exécution du système. Nous abordons deux problèmes d'opacité temporisée : déterminer les temps d'exécution pour lesquels le système est sécurisé (c'est-à-dire pour lesquels l'attaquant ne peut pas déduire d'information confidentielle) et déterminer si un système est sécurisé pour tous ses temps d'exécution possibles.

*Intervenant

Vérification formelle d'une carte à puce pour une certification Critères Communs

Adel Djoudi [1], Martin Hána [1], Nikolai Kosmatov * [2]

[1] THALES [France] – THALES – France
[2] Thales Research and Technology – THALES – France

Ce résumé étendu présente la vérification formelle récente d'une implémentation de machine virtuelle de carte à puce réalisée par Thales à l'aide de la plateforme de vérification Frama-C. Elle a été réalisée en vue d'une certification Critères Communs de niveau EAL6 (pour laquelle le certificat a été délivré en 2021). Les propriétés visées incluent les propriétés de sécurité habituelles, telles que l'intégrité et la confidentialité, qui doivent être assurées par le mécanisme de contrôle d'accès de la machine virtuelle. Ce travail a été initialement publié à FM 2021.

*Intervenant

Illustration de spécifications temporisées paramétrées sur des signaux continus

Etienne André * [1]

[1] Laboratoire Lorrain de Recherche en Informatique et ses Applications – Institut National de Recherche en Informatique et en Automatique, Université de Lorraine, Centre National de la Recherche Scientifique : UMR7503 – France

La spécification de propriétés peut s'avérer délicate. Nous proposons ici une approche automatisée pour l'illustration de propriétés données sous forme d'automates étendus avec des contraintes temporelles et des paramètres temporels, et qui peuvent également spécifier des contraintes sur des signaux à valeurs continues. En d'autres termes, étant donné une telle spécification et un automate bornant le comportement admissible de chacun des signaux, notre approche construit des exécutions continues fournissant ainsi des exemples d'exécutions réelles ou impossibles pour la spécification de départ. Dans notre approche, tant la spécification que les automates bornant les comportements admissibles sont donnés sous forme d'une sous-classe des automates hybrides linéaires, plus précisément des automates temporisés étendus par des vitesses d'horloges arbitraires, des contraintes sur les signaux, et des paramètres temporels ; notre méthode génère alors des exécutions concrètes permettant d'illustrer la spécification.

*Intervenant

POSTER/DEMO

A qualitative pilot for complex systems simulation

Baptiste Gueuziec * [1], Frédéric Boulanger [2], Jean-Pierre Gallois [1]

[1] CEA – CEA-LIST – France
[2] Laboratoire Méthodes Formelles – Institut National de Recherche en Informatique et en Automatique, CentraleSupélec, Université Paris-Saclay, Centre National de la Recherche Scientifique : UMR9021, Ecole Normale Supérieure Paris-Saclay – France

Hybrid systems are complex systems that use discrete and continuous variables, combining several different working modes. There are currently many quantitative methods allowing verification and prediction of such systems. However, they can rapidly become very consuming, especially when the system is not entirely known or when one does not know what to seek. It could be relevant for simplified model checking and prediction to bring the model to a higher level of abstraction to avoid slow and expensive computation and deal with coarse approximations of the system's behavior. This unprecise result may allow us to locate areas of interest or danger in the model and give a direction for a quantitative and more precise simulation adapted to our needs. Some searches have been achieved to simplify this modeling using qualitative reasoning. But although they gave results on dynamic or static systems, they found limits with mixed ones. Moreover, I found no proposition of making the two models communicate. This thesis aims to unify the different qualitative abstraction theories into something more general to create a qualitative pilot that could predict and supervise the execution and simulation of quantitative models.

*Intervenant

Ensuring open and scalable interoperability for smart information system

Boubou Thiam Niang * [1], Giacomo Kahn [1], Nawel Amokrane [2], Yacine Ouzrout [1], Mustapha Derras [2], Jannik Laval [1]

[1] DISP, EA4570, 69676 Bron, France – Univ Lyon, Univ Lumière Lyon 2, INSA Lyon, Université Claude Bernard Lyon 1 – France
[2] Berger-Levrault – Berger-Levrault – France

Information systems (IS) in modern companies must be reactive and able to communicate with third-party IS. It is therefore necessary to establish agile interoperability between information systems. Indeed, the components are designed independently and meet different technical and domain standards that are constantly evolving. Establishing and maintaining interoperability becomes a major challenge, as solutions are developed manually and are not reusable in most cases. This poster presents our work on the automatic generation of interoperability mechanisms, called connectors. Indeed, connectors share common characteristics. For instance, most components use a protocol for communication, encapsulate the information exchanged using data formats. In addition, each of these common characteristics can vary depending on the communication need. With this in mind, we have adopted for software product line engineering to help manage connector variability. Our approach identifies commonalities and variability in connectors from existing interoperability mechanisms implemented in the industry through reverse engineering. We then constructed a feature model to represent the common and variable characteristics of the connectors. These extracted features allowed us to create a connector metamodel that covers all the entities needed to create any connector. This metamodel provides an abstraction level for connectors to make them platform independent. The resulting metamodel of connectors shows that they can be considered as first-class entities, which makes it possible to automatically generate source code for these connectors or to generate a configuration to create connectors from existing subcomponents. The generation of the codes or the configuration of the connectors is done according to the configuration of a valid connector based on the feature model. An example of use case is available on git https://cvs.disp-lab.fr/spl-interop-connector-generation-approach.

*Intervenant

Mitten, A Scenario-Based Consensus Protocols Testing Tool

Çagdas Bozman * [1], Mohamed Iguernlala [1], Michael Laporte [1], Maxime Levillain [1], Alain Mebsout [1], Sylvain Conchon [2]

[1] Functori – Functori – France
[2] Nomadic Labs Univ. Paris Saclay – Nomadic Labs, Univ. Paris Saclay – France

Mitten is a man-in-the-middle proxy between a set of nodes, designed to describe and run tests against consensus protocols implementations. It is configurable to filter and examine network messages according to given scenarios written in a DSL built on OCaml.

Mitten enables writing and simulating subtle cases to reproduce behaviors that are difficult to exhibit under normal circumstances. For instance, we successfully used it to test various corner cases, improvements, and fixes of Tenderbake, the new PBFT consensus protocol of the Tezos blockchain.

*Intervenant

Guaranteeing Timed Opacity using Parametric Timed Model Checking

Dylan Marinho * [1], Etienne André [2], Didier Lime [3], Sun Jun [4]

[1] Université de Lorraine, LORIA – Université de Lorraine, CNRS, LORIA, F-57000 Metz – France
[2] Université de Lorraine, CNRS, Inria, LORIA, Nancy, France – Université de Lorraine – France
[3] LS2N – Nantes Université - École Centrale de Nantes, CNRS : UMR6004 – France
[4] School of Information Systems, Singapore Management University, Singapore – Singapour

Information leakage can have dramatic consequences on systems security. Among harmful information leaks, the timing information leakage occurs whenever an attacker successfully deduces confidential internal information depending on the system execution time. We address the following timed opacity problem: given a timed system, a private location and a final location, synthesize the execution times from the initial location to the final location for which one cannot deduce whether the system went through the private location. We also consider the full timed opacity problem, asking whether the system is opaque for all execution times. We show that these problems are decidable for timed automata (TAs) but become undecidable when one adds parameters, yielding parametric timed automata (PTAs). We then devise an algorithm for synthesizing PTAs parameter valuations guaranteeing that the resulting TA is opaque and finally show that our method can also apply to program analysis.

*Intervenant

ADT4HPC: Algebraic Data Types for High Performance Computing

Thaïs Baudon * [1], Gabriel Radanne [2], Laure Gonnord [3]

[1] LIP – École normale supérieure - Lyon (ENS Lyon), CNRS – France
[2] LIP – École normale supérieure - Lyon (ENS Lyon), CNRS, INRIA – France
[3] Grenoble-INP / LCIS – LCIS, Université Grenoble Alpes, Grenoble-INP – France

Initially present only in functional languages such as OCaml and Haskell, Algebraic Data Types have now become pervasive in mainstream languages, providing nice data abstractions and an elegant way to express functions though pattern-matching. Numerous approaches have been designed to compile rich pattern matching to cleverly designed, efficient decision trees. However, these approaches are specific to a choice of internal memory representation which must accommodate garbage-collection and polymorphism.

ADTs now appear in languages more liberal in their memory representation such as Rust. Notably, Rust is now introducing more and more optimizations of the memory layout of Algebraic Data Types. As memory representation and compilation are interdependent, it raises the question of pattern matching compilation in the presence of non-regular, potentially customized, memory layouts.

In this article, we present Knit&Frog, a framework to compile pattern-matching for monomorphic ADTs, parametrized by an arbitrary memory representation. We propose a novel way to describe choices of memory representation along with a validity condition under which we prove the correctness of our compilation scheme. The approach is implemented in a prototype tool ribbit.

*Intervenant

Model-driven deployment of Digital Twins for Smart Environments - The Human at home projecT case study

Alireza Asvadi [1], Gaëlic Bechu [2], Antoine Beugnard [2], Caroline Cao [2], Christophe Lohr [2], Panagiotis Papadakis [2], Christelle Urtado [3], Quentin Perez * [3], Sylvain Vauttier *

3

[1] Lab-STICC, UMR CNRS 6285 – IMT Atlantique, Brest, France, CNRS : UMR6285 – France
[2] Lab-STICC, UMR CNRS 6285 – CNRS : UMR6285, IMT Atlantique, Brest, France – France
[3] EuroMov - Digital Health in Motion (Euromov DHM) – IMT - MINES ALES, Institut Mines-Télécom [Paris], Université de Montpellier : $UR_UM_IMT_102$ – $-UniversitdeMontpellierUFRSTAPS700avenueduPicSaintLoup34090Montpellier, France$

HUT is a multidisciplinary project that aims at studying the impact of smart environments on well-being. Its main equipment is an observatory apartment that collects data generated by daily life activities of voluntary occupants thanks to a large variety of sensors (temperature, hygrometry, CO2, light, presence, doors, electric and water consumptions, ...)
How to use these datasets for studying the integration of AI components in smart environments? How to continue these experiments after the end of the HUT project?

*Intervenant

Fédération de modèles, une solution d'assemblage de modèles pour l'interopérabilité de sources d'information hétérogènes : l'approche FML / Openflexo

Sylvain Guérin * [1], Joël Champeau [1], Fabien Dagnat [2], Antoine Beugnard [2], Jean-Christophe Bach [3], Salvador Martinez [2]

[1] ensta bretagne – ENSTA Bretagne, Ensta-Bretagne – France
[2] IMT Atlantique – IMT Atlantique – France
[3] IMT Atlantique – IMT Atlantique – France

L'Ingénierie Dirigée par les Modèles (IDM) traite rarement avec de modèles isolés, mais plutôt de collections de modèles, sources d'information et artefacts interconnectés. Les relations entre les modèles sont au coeur de nombreuses activités d'ingénierie orientées modèles, telles que les transformations, les compositions et les liens de traçabilité. Nous proposons ici le paradigme de fédération de modèles, dans lequel modèles et leurs relations sont traités comme un tout. Au delà de l'approche, nous proposons un langage outillé et une plate-forme logicielle permettant la mise en oeuvre de la fédération de modèles.

*Intervenant

A Contract and Facet Based Method for Modelling and Verification of Heterogeneous Systems

A. Abdelkader Khouass * [1,2], J. Christian Attiogbé [1]

[1] LS2N – CNRS : UMR6004, Université de Nantes – France
[2] Université de Tlemcen, LRIT – Algérie

With the continuous expansion of concurrent and distributed systems, Component Based Software Engineering still have a great importance. For constructing these systems which do not have a precise boundary. They are open and made of various specific components built with different languages and environments. The modeling, assembly, analysis and maintenance of such open and complex heterogeneous systems are still challenging in software engineering. The components of heterogeneous systems may cover various concerns, ranging from their nature (software or physical devices), their offered functionalities, to their specific features. The composition of these components should be flexible, and it must be adapted to the reused components and to their local features, whatever the origin of these components.We propose a method for modeling, composing and verifying heterogeneous systems. The method consists in: equipping individual components with generalized contracts that integrate various facets related to different concerns; composing these components according to their facets and verifying the resulting system with respect to the involved facets as well. We illustrate the use of the method with a case study. The proposed method may be used or extended to cover more facets, and by strengthening assistance tool through proactive aspects in modeling, composing multi-facets contracts and finally the verification of the heterogeneous systems.

*Intervenant

Time-Traveling Queries for Faster Debugging and Program Comprehension

Maximilian Ignacio Willembrinck Santander * [1], Steven Costiou [2], Anne Etien [3], Stéphane Ducasse [4]

[1] CRIStAL – Research Centre in Computer Science, Signal and Automatic Control of Lille (CRIStAL UMR 9189) – France
[2] CRIStAL – Univ. Lille, Inria, CNRS, Centrale Lille, UMR 9189 CRIStAL, F-59000 Lille, France – France
[3] Inria Lille - Nord Europe – Institut National de Recherche en Informatique et en Automatique – France
[4] Inria Lille - Nord Europe (Inria Lille - Nord Europe) – Institut National de Recherche en Informatique et en Automatique – Parc Scientifique de la Haute Borne 40, avenue Halley Bât.A, Park Plaza 59650 Villeneuve dÁscq, France

Efficiently debugging a program requires program comprehension. To acquire it, developers explore the program execution, a task often performed using interactive debuggers. Unfortunately, exploring a program execution through standard interactive debuggers is a tedious and costly task. In this poster, we propose Time-Traveling Queries (TTQs) to ease program exploration. TTQs is a mechanism that automatically explores program executions to collect execution data. These data are used to time-travel through execution states, facilitating the exploration of program executions. We built a set of key TTQs based on typical questions developers ask when trying to understand programs. We conducted a user study with 34 participants to evaluate the impact of our queries on program comprehension activities. Results show that, compared to traditional debugging tools, TTQs significantly improve developers' precision, while reducing required time and efforts when performing program comprehension tasks. Link to the demo : https://drive.google.com/file/d/12MlQZO9X1WN7e1LUqm5p0E-x8VJXQz0h/view

*Intervenant

ClassName Distribution Visualization: detecting inconsistencies in class names

Nour Jihene Agouf * [1], Stéphane Ducasse [2], Anne Etien [3]

[1] Inria Lille - Nord Europe – Institut National de Recherche en Informatique et en Automatique – France
[2] Inria Lille - Nord Europe (Inria Lille - Nord Europe) – Institut National de Recherche en Informatique et en Automatique – Parc Scientifique de la Haute Borne 40, avenue Halley Bât.A, Park Plaza 59650 Villeneuve dÁscq, France
[3] Inria Lille - Nord Europe – Institut National de Recherche en Informatique et en Automatique – France

Understanding software starts with understanding the functionalities of its classes. These functionalities are summarized in a simple descriptive class name. Class names should be both correct in the sense that they refer to the exact functionality of the class and, consistent with the system's naming convention. However, not all class names fulfill these criteria, which makes it hard for developers to understand the logic behind thousands of lines of source code. The lack of software understanding leads to an unorganized software evolution. Luckily, with the use of simple techniques such as visualizations, large data can be transformed from an abstract form into visual shapes familiar to the human brain which makes it easy for developers to assess and memorize the logic behind the lines of source code. Indeed, our approach is based on a visualization called 'ClassNames Distribution' which gives an overview of the distribution of classes over packages and helps in detecting inconsistencies in class names from an inheritance perspective. The idea behind inheritance consistent naming is that a hierarchy represents a family of classes having common behavior. This behavior is usually described at the end of the class name when programming in English, other behavior might also emerge therefore a new vocabulaire is used. 'ClassNames Distribution' visualization does not only give the opportunity for developers to have an overview of the system's architecture and the vocabulary used to avoid future naming violations but it is firstly intended to help in detecting suspicious misnaming cases in order to correct them for a better software evolution.

*Intervenant