Actes des journées du GDR GPL 2025

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

Pau, les 16-19 juin 2025

Comité d’organisation

- Mireille Blay-Fornarino
- Jean-Michel Bruel
- Catherine Dubois
- Olivier Le Goaër
- Nicolas Magaud
- Adel Noureddine
- Pascal Poizat
- Responsables des groupes de travail et d’AFADL

Editeurs

- Mireille Blay-Fornarino
- Catherine Dubois
- Nikolaï Kosmatov

Préface

C'est avec grand plaisir que nous vous avons accueillis pour les seizièmes journées nationales du GDR « Génie de la Programmation et du Logiciel »(GPL) qui se sont déroulées à Pau du 16 au 19 juin 2025. Les journées nationales, temps fort de l'activité de notre GDR, donnent l'occasion de nous enrichir des travaux récents présentés, d'échanger entre groupes de travail, de découvrir de nouvelles problématiques et de permettre aux jeunes chercheurs de s'ouvrir à la richesse de notre communauté dans un contexte convivial. Ces journées étaient co-localisées avec la 24ème édition de AFADL, l'atelier francophone sur les Approches Formelles dans l'Assistance au Développement de Logiciels.

Nous avons accueilli, dans l'ordre, les trois conférenciers invités suivants :

— Franck Barbier (Professeur à l'université de Pau) a présenté une conférence intitulée « Économie de la Programmation et du Logiciel » ;
— Frédéric Loulergue (Professeur à l'université d'Orléans), invité partagé avec l'atelier AFADL, a donné une conférence intitulée « Approches formelles pour les composants et outils système » ;
— Federica Sarro (Professeure à University College à Londres) a présenté ses travaux dans un exposé intitulé « Search-Based Software Engineering for Greener and Fairer Artificial Intelligence Systems ».

En plus de ces conférences invitées, les groupes de travail du GDR GPL ont animé onze sessions, dont deux partagées avec l'atelier AFADL. Durant ces journées, un fort accent a été mis sur l'élaboration de nouveaux défis dans les sciences du logiciel. Ainsi des sessions de travail en commun ont été mises en place. Enfin, un atelier plus particulièrement dédié aux doctorants et jeunes docteurs, portant sur les concours de chercheurs et maître.sses de conférences, a été animé par Paul Temple et Catherine Dubois.

Nous avons également reçu la visite d'Anne Siegel, Directrice Adjointe Scientifique au CNRS en charge du suivi des GDRs, qui a présenté quelques nouvelles du CNRS. Nous profitons de cette préface pour la remercier tout particulièrement de son soutien tout au long de notre mandat de directrices du GDR GPL.

Le GDR GPL a à cœur de mettre à l'honneur les jeunes chercheurs. C'est pourquoi nous décernons, depuis 2013, un prix de thèse du GDR. Le prix de thèse GPL 2024 a été attribué à Mickaël Laurent pour sa thèse intitulée « Inférence de types polymorphes pour des langages dynamiques : reconstruction de types pour des systèmes combinant polymorphisme paramétrique, surcharge et sous-typage », préparée à l'université Paris Cité, IRIF et l'université Paris-Saclay, LMF sous la direction de Giuseppe Castagna and Kim Nguyen. Un accessit a été décerné à Anouck Chan pour sa thèse intitulée « Co-conception d'un produit et de son système industriel : une approche ingénierie des exigences pour l'aérospatial », préparée à l'ISAE SupAero et l'ONERA sous la direction de Thomas Polacsek et Stéphanie Roussel. Les lauréats ont présenté leurs travaux lors d'une session plénière dédiée. Félicitations aux lauréats ! Le jury chargé de sélectionner les lauréats a été présidé conjointement par Sandrine Blazy et Pascal Poizat, que nous remercions vivement, ainsi que l'ensemble des membres du jury.

Nous remercions Thomas Degueule qui a assuré cette année encore la responsabilité de la session posters et outils, toujours riche en discussions. Le prix du meilleur poster des journées GPL 2025 a été attribué à Meriem Smati (DISP/INSA Lyon, Poytechnique Montréal) pour son poster intitulé « The use of Digital Twins to Bolster the Resilience of IoT System-of-Systems ». Dix-huit posters dont trois ont également donné lieu à une démonstration, ont été présentés. Chaque poster a fait l'objet d'une présentation en 60 secondes en séance plénière, merci à tous d'avoir accepté ce type d'exercice, avec parfois beaucoup d'humour.

Avant de clôturer cette préface, nous tenons à remercier tous ceux qui ont contribué à l'organisation de ces journées nationales, et tout particulièrement Olivier Le Goaër, Adel Noureddine et les membres de leur comité d'organisation, sans qui ces journées n'auraient pas été possibles, ainsi que les responsables des groupes de travail et les membres du comité de direction du GDR GPL. Nous remercions Nikolai Kosmatov qui a pris en main, une fois encore, les actes des journées nationales.

Mireille Blay-Fornarino et Catherine Dubois
Co-directrices du GDR « Génie de la Programmation et du Logiciel »

Table des matières

Conférences invitées **5**

Approches formelles pour les composants et outils système, Frédéric Loulergue . . 6

Search-Based Software Engineering for Greener and Fairer Artificial Intelligence Systems, Federica Sarro 7

Économie de la Programmation et du Logiciel, Franck Barbier 8

Nouvelles du CNRS **9**

Nouvelles du CNRS, Anne Siegel 10

Prix de Thèse **11**

Prix de thèse, Pascal Poizat 12

Atelier préparation concours **13**

Atelier Préparation aux concours, Paul Temple [et al.] 14

GT IDM **15**

Intégration d'Exigences Exécutables dans le Prototypage, Yann Le Goff [et al.] . 16

10 years of Model Federation with Openflexo: Challenges and Lessons Learned, Jean-Christophe Bach [et al.] 17

Continuous Secure Evolution of Software Models, Chahrazed Boudjemila [et al.] . 18

BridgeSec: Facilitating effective communication between security engineering and systems engineering, Nan Messe [et al.] 19

GT GLIA **20**

Présentation du projet FATES-MLOps, Jean-Michel Bruel 21

Présentation du projet WestOps, Paul Temple 22

Présentation du projet AdaptiveMLOps, Christelle Urtado 23

Présentation du projet PROFIL, Nicolas Lacroix 24

GT DevX **25**

Gamifying model-based engineering: The PapyGame tool, Jean-Michel Bruel . . 26

DevX : introduction du groupe de travail, Steven Costiou [et al.] 27

Empirically Evaluating the Impact of Object-Centric Breakpoints on the Debugging of Object-Oriented Programs, Steven Costiou [et al.] 28

On Polyglot Program Testing, Benoit Combemale [et al.] 29

What the Fix? A Study of ASATs Rule Documentation, Jean-Rémy Falleri . . . 30

GT Logiciel Eco-Responsable **31**

Prédiction de bornes paramétriques de consommation énergétique : point de vue et perspectives de l'interprétation abstraite, Louis Rustenholz 32

A quel point les assistants de code comme GitHub Copilot consomment-ils de l'énergie?, Tristan Coignion 33

AndroWatts: Unpacking the Power Consumption of Mobile Device's Components, Édouard Guégain 34

Investigating the Impact of Software Design Patterns on Energy Consumption, Adel Noureddine [et al.] 35

GT VL **36**

Exploring Performance of Configurable Software Systems: the JHipster Case Study, Édouard Guégain 37

Does Functional Package Management Enable Reproducible Builds at Scale? Yes, Théo Zimmermann 38

Roseau: Fast, Accurate, Source-based API Breaking Change Analysis in Java, Corentin Latappy 39

Fingerprinting and Building Large Reproducible Datasets, Romain Lefeuvre . . . 40

GT GLSec **41**

A Model-based Approach for Assessing the Security of Cyber-Physical Systems, Dominique Blouin 42

Gestion automatisée des consentements pour la protection des données personnelles, Selena Lamari 43

Lutte Informatique d'Influence - Une approche conceptuelle pour la détection de Fakes News, Nicolas Belloir 44

GT IE + HIFI **45**

Exigences de qualité extra-fonctionnelles des LLMs, Jean-Michel Bruel 46

Leveraging Verification to Enhance Formal Explainable AI for Neural Networks, Ryma Boumazouza [et al.] 47

Discussion, Sylvain Conchon [et al.] 48

GT CLAP + YODA **49**

Simulations et compilation vérifiée, Nicolas Chappe 50

Vérification formelle de propriétés interactives: retours d'expérience, Cécile Marcon 51

GT MTV2 - AFADL **52**

Apprentissage et test pour les machines à états finis temporisées avec délais de sortie, Evgenii Vinarskii [et al.] 53

Experimental Evaluation of LLMs for Test Generation, Tiago Costa [et al.] . . . 54

Projet ASMA VéDySec : Vérification Dynamique de Propriétés de Sécurité, Nikolai Kosmatov [et al.] 55

Projet ANR RAPID VVaMIA - Vérification et Validation des systèmes embarqués augmentés par les Modèles et l'IA générative, Frédéric Dadeau [et al.] 56

GT LVP - AFADL **57**

BICOQ : une formalisation des bigraphes dans Coq, Cécile Marcon [et al.] 58

Une sémantique mécanisée d'un langage FRP avec effets, Jordan Ischard [et al.] . 59

Prouvez vos coloriages: vérification formelle du coloriage de cache de l'hyperviseur Bao, Axel Ferréol [et al.] 60

Langage Chips : Modélisation et contrôle de systèmes distribués à base de composants, Anna Gallone 61

AFADL **62**

Découverte de processus probabiliste avec des arbres de processus stochastiques, Pierre Cry 63

Tail Modulo Async-Await, Emma Nardino [et al.] 64

Model Checking de LTL sur Traces Finies et Infinies avec Domaines Concrets, Julien Brunel [et al.] 65

VASSAL Project: Verification and Analysis for Safety and Security of Applications in Life, Julien Signoles [et al.] 66

Approche neuro-symbolique pour générer un référentiel de test, Eléa Jacquin . . 67

Transformations automatisés de preuves Coq, Alexandre Jean [et al.] 68

Langage de spécification de contexte pour vérifier formellement des propriétés de consentement sur des modèles et du code, Myriam Clouet [et al.] 69

Project & Conquer : Élimination de quantificateurs pour la vérification de problèmes d'accessibilité dans les réseaux de Petri, Nicolas Amat [et al.] 70

Posters et démos **71**

Conférences invitées

Approches formelles pour les composants et outils système

Frédéric Loulergue * [1]

[1] LIFO – Université d'Orléans, INSA Centre Val de Loire – France

Cet exposé présente des approches formelles d'une part pour la vérification de composants système et d'autre part pour la formalisation d'outils système. Dans la première partie, on considérera des programmes C, tandis que dans la seconde, on considérera des langages de configuration.

Frama-C est un framework pour l'analyse et la vérification de programmes C. Il propose une collection d'analyses sous forme de greffons qui peuvent collaborer, parmi lesquels EVA pour l'analyse statique par interprétation abstraite, WP pour la vérification déductive, E-ACSL pour la vérification dynamique, RTE pour la génération d'annotation pour assurer l'absence de Runtime Error ou encore MetACSL pour la vérification de propriétés de haut-niveau. Nous aborderons les fonctionnalités de Frama-C via des cas d'études de composants systèmes avec un focus sur la vérification déductive, mais aussi ses limites et quelques recherches en cours pour les repousser, notamment dans le cadre du projet ANR CoMeMoV.

Les grands systèmes logiciels distribués sont désormais omniprésents. Les systèmes à base de composants offrent un moyen pratique de structurer de grands systèmes, en particulier les systèmes déployés dans le Cloud, au cœur ou à la périphérie du réseau. Les opérations DevOps, fournies par divers outils, qui incluent la configuration et la reconfiguration de système, sont nécessaires pour gérer divers types de scénarios tels que la tolérance aux pannes, l'évolutivité, les mises à jour logicielles ou diverses optimisations. De tels changements peuvent entraîner des incidents. Le principal avantage des outils DevOps est leur intégration et leur adoption dans la communauté DevOps. Leur inconvénient est qu'ils manquent de spécifications formelles et mêmes textuelles. L'exposé présentera nos premiers efforts, dans le contexte du projet ANR ForCoaLa, pour améliorer la compréhension de ces langages en se basant sur des sémantiques formelles mécanisées.

*Intervenant

Search-Based Software Engineering for Greener and Fairer Artificial Intelligence Systems

Federica Sarro * [1]

[1] University College, London – Royaume-Uni

As artificial intelligence (AI) systems become increasingly embedded in our daily lives, the need for sustainable and equitable development practices has never been more urgent. In this keynote, I explore how Search-Based Software Engineering (SBSE) can be harnessed to address two of the most pressing challenges in modern AI: environmental sustainability and algorithmic fairness.

Drawing on over a decade of research at the intersection of software engineering, optimization, and machine learning, I present a compelling vision for how SBSE techniques-such as evolutionary algorithms and multi-objective optimization-can be used to automatically configure, adapt, and improve AI systems with minimal environmental impact and maximal social benefit. I will showcase recent advances in green AI, including energy-aware model selection and deployment strategies, as well as fairness-aware optimization frameworks that balance performance with ethical considerations. Through real-world case studies and empirical evidence, this talk will demonstrate how SBSE not only enhances the efficiency and effectiveness of AI development but also provides a principled foundation for building AI systems that are both greener and fairer.

*Intervenant

Économie de la Programmation et du Logiciel

Franck Barbier * [1]

[1] Laboratoire Informatique de l'Université de Pau et des Pays de l'Adour – Université de Pau et des Pays de l'Adour, Université de Pau et des Pays de l'Adour : EA3000 – France

Notre activité de recherche est censée (pour partie) mener à du transfert voire de la valorisation par la création d'une start-up par exemple. La frontière entre recherche et innovation est souvent difficile à cerner car notre recherche se décline sur l'échelle : théorique, fondamentale et appliquée. Pour la Programmation et le Logiciel, notre communauté est "au coeur du réacteur numérique". Il semble ainsi intéressant d'analyser la façon dont nous pouvons, si désiré, faire bouger la société et l'économie à partir de nos résultats de recherche.

A travers une expérience personnelle, cette keynote tente de donner les tenants et aboutissants du transfert et de la valorisation de recherches sur la Programmation et le Logiciel. Quelques clefs s'y ajoutent pour les chercheurs s'interrogeant sur une telle "mission". Nous sommes susceptibles de (attendus pour ?) établir les nouveaux mécanismes et les nouvelles modalités de conception des applications informatiques. Sobriété énergétique, respect de la vie privée, calcul cloud/edge, IA... sont les sujets prégnants du moment. Voulons-nous (réellement) influer sur cela ?

*Intervenant

Nouvelles du CNRS

Nouvelles du CNRS

Anne Siegel * [1]

[1] Institut sciences informatiques – CNRS – France

Anne Siegel présente les nouvelles du CNRS et répond aux questions des participants.

*Intervenant

Prix de Thèse

Prix de thèse

Pascal Poizat * [1]

[1] LIP6 – U. Paris-Nanterre, University Paris Sorbonne – France

Créé en 2013 pour récompenser chaque année une excellente thèse préparée au sein du GDR GPL, le **Prix de thèse du GDR GPL** a pour objectif de promouvoir les travaux du GDR GPL auprès de la communauté informatique.

Le prix est décerné par un jury couvrant les thématiques du GDR GPL. Pour l'édition concernant les thèses soutenues en 2024, le jury est présidé par Sandrine Blazy et Pascal Poizat.

Le prix de thèse GPL 2024 est attribué à Mickaël Laurent pour sa thèse intitulée "Polymorphic type inference for dynamic languages: Reconstructing types for systems combining parametric, ad hoc, and subtyping polymorphism" préparée à l'Université Paris Cité, IRIF et l'Université Paris-Saclay, LMF sous la direction de Giuseppe Castagna and Kim Nguyen. Lien ver la thèse : https://theses.fr/2024UNIP7076

L'accessit est attribué à Anouck CHAN pour sa thèse intitulée "Co-conception d'un produit et de son système industriel : Une approche ingénierie des exigences pour l'aérospatial" préparée à l'ISAE SupAero et l'ONERA sous la direction de Thomas Polacsek et Stéphanie Roussel. Lien ver la thèse : https://theses.fr/2024ESAE0042

Félicitations aux lauréats!

*Intervenant

Atelier préparation concours

Atelier Préparation aux concours

Paul Temple * [1], Catherine Dubois [2]

[1] Irisa – Univ Rennes, CNRS, INRIA – France
[2] Ecole Nationale Supérieure dÍnformatique pour lÍndustrie et lÉntreprise – ENSIIE, Samovar – France

Le mardi 17 Juin 2025, en fin de journée, s'est déroulé l'atelier dédié aux jeunes chercheurs et chercheuses afin de discuter et de répondre aux questions concernant la préparation aux concours de recrutement (que ce soit au CNRS, à l'Inria ou aux postes de maitres et maîtresses de conférences). Ce moment a regroupé une quinzaine de personnes curieuses d'en apprendre plus sur les éléments permettant de préparer au mieux leurs dossiers de candidature ainsi que de mettre toutes les chances de leur côté pour s'engager pleinement dans cette période difficile mais charnière dans leur future carrière. Cet atelier se fait en complément des deux soirées en ligne qui se déroulent habituellement entre octobre et novembre et qui rentrent plus en détail dans chacun des concours. Ici, le but est d'amorcer la réflexion de potentiels futurs candidats et candidates en présentant brièvement le déroulé des concours mais aussi en prévoyant un moment d'échange long, permettant de répondre aux questions des participants et participantes. Ces échanges se font aussi avec l'aide de nouveaux recrutés et nouvelles recrutées. En l'occurence, cette année, un seul nouveau recruté a pu participer à cet atelier.

Le déroulé de cet atelier "Les concours, parlons en !" a suivi le programme suivant : tout d'abord, Paul et Catherine ont présenté dans les grandes lignes les éléments importants aux concours (la notion de section, les endroits qui proposent des recrutements, un calendrier approximatif des périodes de concours synchronisés (notamment avec une chronologie CNRS -> Inria -> Mcf via Odyssée et la campagne synchronisée), quelques mots sur la qualification, et enfin la préparation du dossier et des auditions). Ensuite, des intervenants issus de l'ENAC et de l'ONERA ont complété cette présentation en détallant les processus de recrutement dans ces deux instituts. Le nouveau recruté présent a partagé son expérience des concours ainsi que les éléments auxquels il faut se préparer pour ses premières années en poste. Finalement, les participants et participantes ont posé leurs questions auxquelles les différents intervenants ont tenté de répondre du mieux possible.

Le support à la première présentation est disponible à l'adresse suivante : https://gdr-gpl.cnrs.fr/wp-content/uploads/2025/06/soiree_concours_GDR-GPL_2025.pdf

*Intervenant

GT IDM

Intégration d'Exigences Exécutables dans le Prototypage

Yann Le Goff * [1,2], Eric Le Pors [3], Pierre Laborde [4], Mickaël Kerboeuf [5], Alain Plantec [5]

[1] Laboratoire des sciences et techniques de l'information, de la communication et de la connaissance – Université de Brest – France
[2] Thales DMS France, SAS – Thales (France) – France
[3] Thales DMS France, SAS – Thales (France) – France
[4] Thales DMS France, SAS – Thales (France) – France
[5] Laboratoire des sciences et techniques de l'information, de la communication et de la connaissance – Université de Brest – France

Chez Thales DMS, le processus de création d'application implique plusieurs équipes. L'équipe d'ingénierie des systèmes produit un ensemble d'exigences que l'application finale doit respecter. La mise en œuvre d'une application conformément à ces exigences peut être très fastidieuse et chronophage. Pour faciliter le développement de l'application finale, un prototype peut être mis en œuvre en tant que représentation intermédiaire du produit final.

L'une des principales difficultés réside dans le lien entre les exigences et les éléments du prototype pour garantir qu'ils sont pris en compte tout au long du développement.

Nous présentons une façon d'utiliser les modèles d'exigences pour vérifier la non régression des exigences dans le prototype pendant sa mise en œuvre grâce à l'exécution de Tracing Points. Nous avons implémenté les Tracing Points dans Pharo et nous expliquons comment les Tracing Points sont intégrés et exécutés dans le prototype pour vérifier les exigences correspondantes.

*Intervenant

10 years of Model Federation with Openflexo: Challenges and Lessons Learned

Jean-Christophe Bach [1], Antoine Beugnard [1], Joël Champeau [2], Fabien Dagnat [3], Sylvain Guérin * [3], Salvador Martinez [4]

[1] Laboratoire des sciences et techniques de l'information, de la communication et de la connaissance – IMT Atlantique – France
[2] Laboratoire des sciences et techniques de l'information, de la communication et de la connaissance – École Nationale Supérieure de Techniques Avancées – France
[3] Laboratoire des sciences et techniques de l'information, de la communication et de la connaissance – IMT Atlantique – France
[4] Laboratoire des sciences et techniques de l'information, de la communication et de la connaissance – IMT Atlantique – France

In the context of complex system development, heterogeneous modeling responds to the need to integrate several domains. This need requires the use of the most appropriate formalism and tooling for each domain to be efficient. Model federation promotes the semantic interoperability of heterogeneous models by providing the means to reify correspondences between different model elements, add custom behaviors and bridge the gap between technological spaces. As such, it can be used as an infrastructure to address many different system engineering problems. This is what we have been doing for over a decade, as part of a close collaboration between a small software engineering startup and academia. This paper reports on this experience. Concretely, we discuss the context, ambitions, and challenges that led to the inception of our practice of model federation, and we present five use cases experiences, stemming from real industrial and academic needs, and elaborate on lessons learned. In addition, we also report on challenges and lessons learned regarding the development and maintenance of a model-driven model federation tool, the Openflexo framework. Finally, we set up a road map for the future of model federation and Openflexo.

*Intervenant

Continuous Secure Evolution of Software Models

Chahrazed Boudjemila * [1], Salvador Martinez [1], Fabien Dagnat [1]

[1] Laboratoire des sciences et techniques de l'information, de la communication et de la connaissance – IMT Atlantique – France

Model-driven Engineering promotes the development of complex systems through the use of a set of models. These models conform to different modeling languages and are built with various tools, leading to a multi-model system. Since these models represent the system under development, dependencies often exist between them. However, this multi-model system evolves to accommodate new requirements, bug fixes, etc. These changes represent a challenge w.r.t. their consistency and is even more critical when dealing with security requirements. Indeed, following the principle of "security-by-design", security has been integrated within models from the early phases of development. Consequently, it also needs to be considered when the models evolve to avoid potential security issues. This thesis proposes a methodology for maintaining security consistency between models. It leverages the model federation paradigm to establish explicit correspondences between meta-model elements and equip them with security rules. These rules can be (re)evaluated when models change, providing an assessment of whether a modification impacts security consistency. The proposed methodology has been successfully applied to several case studies and implemented using existing tools.

*Intervenant

BridgeSec: Facilitating effective communication between security engineering and systems engineering

Nan Messe * [1], Avi Shaked [2]

[1] Smart Modeling for softw@re Research and Technology – Institut de recherche en informatique de toulouse, Université Toulouse Jean Jaurès – France
[2] University of Oxford – Royaume-Uni

We increasingly rely on systems to perform reliably and securely. Therefore, it is imperative that security aspects are properly considered when designing and maintaining systems. However, achieving the security by design ideal is challenging. Security information is typically unstructured, dispersed, hard to communicate, and its assessment is somewhat subjective and tacit. Additionally, the inclusion of security information within design requires integrating the efforts of two knowledge-intensive disciplines: security engineering and systems engineering. In this talk, we introduce BridgeSec, a novel conceptual information-exchange interface to systemise the communication of security information between these two disciplines. The main contribution of BridgeSec lies in its explicit identification of concepts related to vulnerability management, which allows systems engineering and security engineering teams to codify pertinent information. The disciplines involved in the system design can thus coordinate policies, implementations and, ultimately, the security posture. Furthermore, based on the newly unveiled interface, an automated reasoning mechanism is specified. This mechanism allows to reason about the vulnerability posture of systems in a scalable and systematic way. First, we describe and formalise the information-exchange interface BridgeSec and how it can be used to reason about the security of systems designs. Next, we present an open-source prototype – integrated into a threat modelling tool – which rigorously implements the interface and the reasoning mechanism. Finally, we detail two diverse and prominent applications of the interface for communicating security aspects of systems designs. These applications show how BridgeSec can rigorously support the design of systems' security in two representative scenarios: in coordinating security features and policy during design, and in coordinating mitigation to disclosed implementation vulnerabilities.

*Intervenant

GT GLIA

Présentation du projet FATES-MLOps

Jean-Michel Bruel * [1]

[1] Smart Modeling for softw@re Research and Technology – Institut de recherche en informatique de toulouse – France

The international shift from 'Model-centric AI' to 'Data-Centric AI' underscores the need to verify and justify that systems adhere to FATE properties throughout the integration of machine learning (ML) components. Originating from the FAT/ML initiative in 2014, Fairness, Accountability, Transparency, and Ethics (FATE) properties were extended with Security/Safety considerations by the Microsoft FATE research group and more recently embraced by the Data for Good movement at Columbia University. Various algorithms have been developed to tackle different facets of F, T, and S properties, while others rely more on commitment to A and E principles. International regulations and governmental bodies are increasingly proposing frameworks to aid organizations and individuals in promoting responsible design, development, deployment, and usage of AI systems over time. However, to our knowledge, a systematic study or guidance is needed to assist ML scientists and engineers in tracking FATEs properties. Their integration occurs at various stages, from model training to deployment and production use, and varies depending on the application context. This doctoral project systematically explores the integration of FATEs properties justification within software systems incorporating ML components. It addresses scientific challenges by formalizing these properties across contexts and systematically integrating them into the MLOps process. We begin with a cross-referenced analysis of the literature on FATEs and architectural approaches to validating, enriching, and justifying development processes. To circumscribe the scope of the thesis, we will focus on ML model training workflows (e.g., for oversampling under-represented groups), pipeline compositions in LangChain (e.g., to introducing debiasing steps on reinforcement data), and CI/CD workflows (e.g., deploying a parallel explanation model alongside the system). We will focus on fairness, accountability, and transparency, for which various software solutions exist. We will indirectly address ethics and security when associated with specific use cases, as they require dedicated theses and extend to analysis phases and ancillary software systems. Expected outcomes encompass practical implementations, particularly within code generation tools (e.g., investigating StarCoder to generate more FATE-compliant code in collaboration with the Hugging Face development team). Overall, our approach aims to facilitate the adoption of FATE principles by characterizing these properties in ML workflows and developing automated justifications to track and document the compromises and verifications made. In short, our approach aims to formalize FATEs properties, systematically integrate them into the development process, and apply them in real MLOps applications to ensure more responsible and reliable systems.

*Intervenant

Présentation du projet WestOps

Paul Temple * [1]

[1] Institut de Recherche en Informatique et Systèmes Aléatoires – Université de Rennes, Institut National des Sciences Appliquées - Rennes, Université de Bretagne Sud, École normale supérieure - Rennes, Institut National de Recherche en Informatique et en Automatique, CentraleSupélec, Centre National de la Recherche Scientifique, IMT Atlantique – France

Continuous Monitoring and Improvement for Trustworthy AI: A Case Study on content classifiers – WestOps

Ouest France, a leading media company, uses Machine Learning (ML) techniques for content classification, information extraction, and recommendation. Ouest-France is exploring new approaches for MLOps implementation to streamline the evaluation process, ensure timely detection of performance variations, and enable faster and more reliable adaptation of ML components.

The WestOps project aims to address these challenges. Firstly, it aims to simplify integrating functional non-regression tests for classifiers and recommender systems. This will be achieved through a custom domain-specific language that seamlessly integrates within Ouest France's existing ML system. Secondly, the project focuses on early detection of model deviations using variability techniques that incorporate journalist validation. By analyzing results from various models, the project aims to identify and address performance issues. Furthermore, a monitoring module will be developed to raise alerts and trigger feedback loops, further improving the maturity of the MLOps pipeline.

The expected outcomes of WestOps include a global MLOps framework validated through the Ouest France case study, an open-source unified test framework for ML models, and the implementation of performance monitoring and feedback loops in a real-world setting.

The collaboration between Ouest-France, Kereval, and Inria DiverSE presents a groundbreaking opportunity to strengthen the reliability of AI systems for industrial partners. This project's focus is the development of modular MLOps components, designed for adaptation across diverse sectors. By promoting this adaptability, the project will stimulate significant improvements in global machine learning practices and applications.

*Intervenant

Présentation du projet AdaptiveMLOps

Christelle Urtado * [1]

[1] IMT - MINES ALES – Institut Mines-Télécom [Paris] – France

Etendant les principes du DevOps aux Data Sciences et au Machine Learning, le MLOps outille les processus d'entraînement, de déploiement et de maintien en conditions opérationnelles de modèles d'IA intégrés à des architectures logicielles. Un élément clé de ces processus est la gestion de l'entraînement continu de ces modèles d'IA, pour les adapter aux changements observables dans les données qui les alimentent en contexte de production (data et concept drifts). Une solution no-code / low code semble adaptée à la multiplicité des profils des développeurs des modèles d'IA (ingénieur data, IA ou logiciel). Mais sa conception se heurte à la diversité et à la variabilité des éléments constituant les processus de MLOps. Nous proposons ainsi d'étudier comment les concepts de l'ingénierie de domaine (modèles de features, lignes de produits) permettraient de cataloguer les éléments récurrents des processus de MLOps et de documenter les bonnes pratiques de leur mise en œuvre. La connaissance métier ainsi recueillie et formalisée permettra de guider et d'assister la conception des pipelines de MLOps. Elle sera mise en œuvre par une approche d'ingénierie dirigée par les modèles qui proposera un DSL générique et extensible, pour couvrir la variété des éléments des pipelines et des environnements cibles, abstrait (low/no code), pour être facilement utilisable sans expertise mais sans limiter les possibilités de personnalisation et de fine-tuning, et pivot pour supporter les transformations vers les langages de Plateforme/Infrastructure as Code utilisés dans les déploiements. Nous étudierons en particulier la gestion automatique et dynamique du (re)-déploiement de différentes portions du pipeline chez des fournisseurs multiples afin d'optimiser les performances, le coût et l'empreinte environnementale de son exécution et des composants d'IA associés. L'ensemble des solutions sera prototypé et validé (au travers de PoC en utilisant la plateforme du partenaire industriel du projet (fournisseur de solutions cloud) sur des cas d'études concrets fournis au partenaire industriel par de grands groupes français.

*Intervenant

Présentation du projet PROFIL

Nicolas Lacroix * [1]

[1] Scalable and Pervasive softwARe and Knowledge Systems – Laboratoire d'Informatique, Signaux, et Systèmes de Sophia Antipolis – France

Caractérisation, Vérification et Mutualisation des Workflows de ML – PROFIL.

La démocratisation du Machine Learning (ML) à de nombreux domaines d'applications a pour conséquence une profusion de codes mis en commun sans réel contrôle. En pratique, l'expertise portée par ces codes est largement réutilisée pour guider de nouvelles explorations. Cependant, il reste aujourd'hui très difficile d'une part de systématiquement analyser les variations entre les workflows (WF) (i.e. les ensembles partiellement ordonnés de tâches) portés par ces codes, que ce soit pour les réutiliser ou en extraire de nouvelles connaissances et d'autre part de vérifier leur qualité au moins pour ne pas reproduire des erreurs connues.

Dans ce projet, nous adoptons une approche de génie logiciel (GL) pour aborder cette problématique en proposant de lier ingénierie des modèles (au sens GL), analyse statique et statistique pour caractériser ces workflows de ML par des modèles (aussi au sens GL), désormais nommés profils pour éviter la confusion. Plus précisément, notre projet vise à explorer trois aspects complémentaires : (i) quelles informations peut-on et doit-on extraire automatiquement de code ML pour construire un profil permettant son analyse ? (ii) Est-il possible d'identifier systématiquement des erreurs types à partir du profil et d'identifier de mauvaises pratiques ? (iii) Peut-on exploiter la profusion des codes ML pour accélérer la recherche en ML en favorisant, à partir des profils extraits, une mutualisation des connaissances et l'élicitation de bonnes ou mauvaises pratiques ?

Notre hypothèse de recherche est qu'il est possible d'identifier finement les étapes d'un WF en fonction soit des fonctions de bibliothèques logicielles appelées, soit d'informations contextuelles (ex: données en paramètre, appels de fonction précédents). Avec PROFIL, nous fournirons la première plateforme d'analyse de code de ML à des fins de caractérisation de profils en nous appuyant sur un méta-modèle de WF, de la logique propositionnelle, de l'analyse de code et du clustering.

*Intervenant

GT DevX

Gamifying model-based engineering: The PapyGame tool

Jean-Michel Bruel * [1]

[1] Smart Modeling for softw@re Research and Technology – Institut de recherche en informatique de toulouse – France

Modeling is an essential and challenging activity in any engineering environment, and it requires some hard-to-train skills such as abstraction and communication. This makes it difficult for educators to teach or train their students, co-workers, or users. The audience of this paper is both educators and learners who struggle with modeling. To address this challenge, we present PapyGame, a gamified version of a robust modeling environment (Papyrus) that aims to improve the learner's motivation, make the learning process an enjoyable experience, and boost learning outcomes. Gamification is the exploitation of gaming mechanisms for serious purposes, such as promoting behavioral changes, soliciting participation and engagement in activities, and more. The paper presents PapyGame's functionalities, architectures, illustrative scenarios, and its potential impact on both educators and learners.

*Intervenant

DevX : introduction du groupe de travail

Steven Costiou * [1], Valentin Bourcier [1], Benoit Combemale [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, Centrale Lille, Université de Lille, Centre National de la Recherche Scientifique – France
[2] Diversity-centric Software Engineering – Centre Inria de l'Université de Rennes, LANGAGE ET GÉNIE LOGICIEL – France

Le GT DevX fait suite au GT Debugging, en élargissant les thèmes, problématiques et sujets scientifiques et technologiques couverts.

Dans cette session introductive, nous présenterons et résumerons les motivations derrière la création du GT DevX, ainsi que les différentes actions et journées de travail déjà menées lors de cette étape de création.

Nous introduirons ensuite le programme de la session.

*Intervenant

Empirically Evaluating the Impact of Object-Centric Breakpoints on the Debugging of Object-Oriented Programs

Steven Costiou * [1], Valentin Bourcier *

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

Debugging consists in understanding the behavior of a program to identify and correct its defects. Breakpoints are the most commonly used debugging tool and aim to facilitate the debugging process by allowing developers to interrupt a program's execution at a source code location of their choice and inspect the state of the program.
Researchers suggest that in systems developed using object-oriented programming (OOP), traditional breakpoints may be a not effective method for debugging. In OOP, developers create code in classes, which at runtime are instantiated as object-entities with their own state and behavior that can interact with one another. Traditional breakpoints are set within the class code, halting execution for every object that shares that class's code. This leads to unnecessary interruptions for developers who are focused on monitoring the behavior of a specific object. As an answer to this challenge, researchers proposed *object-centric debugging*, an approach based on debugging tools that focus on objects rather than classes. In particular, using *object-centric breakpoints*, developers can select specific objects (rather than classes) for which the execution must be interrupted. Even though it seems reasonable that this approach may ease the debugging process by reducing the time and actions needed for debugging objects, no research has yet verified its actual impact.
To investigate the impact of object-centric breakpoints on the debugging process, we devised and conducted a controlled experiment with 81 developers who spent an average of 1 hour and 30 minutes each on the study. The experiment required participants to complete two debugging tasks using debugging tools with vs. without object-centric breakpoints. We found no significant effect from the use of object-centric breakpoints on the number of actions required to debug or the effectiveness in understanding or fixing the bug. However, for one of the two tasks, we measured a statistically significant reduction in debugging time for participants who used object-centric breakpoints, while for the other task, there was a statistically significant increase. Our analysis suggests that the impact of object-centric breakpoints varies depending on the context and the specific nature of the bug being addressed. In particular, our analysis indicates that object-centric breakpoints can speed up the process of locating the root cause of a bug when the bug can be replicated without needing to restart the program. We discuss the implications of these findings for debugging practices and future research.

*Intervenant

On Polyglot Program Testing

Benoit Combemale * [1], Philémon Houdaille *

[1] Diversity-centric Software Engineering – Centre Inria de l'Université de Rennes, LANGAGE ET GÉNIE LOGICIEL – France

In modern applications, it has become increasingly necessary to use multiple languages in a coordinated way to deal with the complexity and diversity of concerns encountered during development. This practice is known as polyglot programming. However, while execution platforms for polyglot programs are increasingly mature, there is a lack of support in how to test polyglot programs. This paper is a first step to increase awareness about polyglot testing efforts. It provides an overview of how polyglot programs are constructed, and an analysis of the impact on test writing at its different steps. More specifically, we focus on dynamic white box testing, and how polyglot programming impacts selection of input data, scenario specification and execution, and oracle expression. We discuss the related challenges in particular with regards to the current state of the practice. We envision in this paper to raise interest in polyglot program testing within the software engineering community, and help in defining directions for future work.

*Intervenant

What the Fix? A Study of ASATs Rule Documentation

Jean-Rémy Falleri * [1]

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

Automatic Static Analysis Tools (ASATs) are widely used by software developers to diffuse and enforce coding practices. Yet, we know little about the documentation of ASATs, despite it being critical to learn about the coding practices in the first place. We shed light on this through several contributions. First, we analyze the documentation of more than 100 rules of 16 ASATs for multiple programming languages, and distill a taxonomy of the purposes of the documentation—What triggers a rule; Why it is important; and how to Fix an issue—and its types of contents. Then, we conduct a survey to assess the effectiveness of the documentation in terms of its goals and types of content. We highlight opportunities for improvement in ASAT documentation. In particular, we find that the Why purpose is missing in half of the rules we survey; moreover, when the Why is present, it is more likely to have quality issues than the What and the Fix.

*Intervenant

GT Logiciel Eco-Responsable

Prédiction de bornes paramétriques de consommation énergétique : point de vue et perspectives de l'interprétation abstraite

Louis Rustenholz *† [1,2]

[1] Universidad Politecnica de Madrid (UPM) – Espagne
[2] IMDEA Software Institute – Espagne

Les techniques d'analyse statique, comme l'interprétation abstraite, proposent de découvrir automatiquement certaines propriétés des programmes et systèmes, et d'en borner les comportements. En particulier, elles offrent maintenant la perspective de prédire des bornes de la consommation énergétique d'un programme, dont le code source et environnement d'exécution sont donnés, en une unique analyse couvrant l'ensemble infini des entrées et interactions possibles. Les bornes obtenues sont paramétriques : ce sont des fonctions (d'une abstraction) de l'entrée et de l'environnement, possiblement dépendantes d'autres paramètres laissés libres par le développeur. Dans cet exposé, nous présenterons quelques avancées récentes de ce domaine de recherche, et esquisserons l'état des connaissances sur l'analyse automatique de la consommation en ressources (temps, mémoire, énergie, etc.), avant de discuter des applications potentielles de ces techniques aux enjeux de la sobriété numérique, en les replaçant dans le paysage plus large des approches en faveur d'une informatique éco-responsable. Nous en profiterons pour mettre en avant plusieurs niveaux d'action complémentaires des sciences du logiciel : des optimisations de compilateur traquant les gâchis de ressources, à la transparence énergétique fournissant des leviers aux développeurs, et peut-être même aux législateurs.

*Intervenant
†Auteur correspondant: louis.rustenholz@imdea.org

A quel point les assistants de code comme GitHub Copilot consomment-ils de l'énergie?

Tristan Coignion * [1]

[1] Inria Lille - Nord Europe – Université de Lille - Sciences et Technologies – France

Les assistants de code tels que GitHub Copilot sont de plus en plus populaires auprès des développeurs. Mais la génération des réponses étant faites dans le cloud, les utilisateurs ne peuvent pas connaître le réel impact énergétique de leur assistant de code. Nous avons donc mesuré la consommation énergétique d'assistants de code, en reproduisant leur fonctionnement sur nos serveurs. Nous avons vu que les assistants de code consomment beaucoup d'énergie, et qu'il y a beaucoup de pistes d'améliorations pour réduire leur consommation.

*Intervenant

AndroWatts: Unpacking the Power Consumption of Mobile Device's Components

Édouard Guégain * [1]

[1] R&D GREENSPECTOR – Greenspector – France

Power efficiency is crucial for mobile devices, where software inefficiencies can rapidly drain battery life and reduce device longevity. To help developers diagnose inefficiency issues and optimize power use, this paper introduces a novel methodology for estimating power consumption at a hardware component level in mobile devices. Unlike existing approaches that rely on coarse-grained battery discharge measurements, our approach consists of modeling per-component power usage as an inverse problem, utilizing linear statistical models and system metrics to estimate the contributions of individual components. In this paper, we model the total power usage using system metrics, we evaluate the accuracy of per-component power estimations, and we investigate the impact of dataset size on the model's performance. Our empirical evaluation shows that linear models achieve high predictive accuracy, with R-squared values exceeding 0.90 for total power usage, and robust correlations are observed between predicted and ground-truth power consumption for key components such as the CPU and GPU. We also observe that the amount of measures required to build such a model is of the order of magnitude of the hundred rather than the thousand.

*Intervenant

Investigating the Impact of Software Design Patterns on Energy Consumption

Adel Noureddine * [1], Olivier Le Goaer *

[2]

[1] IUT des Pays de l'Adour – Laboratoire d'Informatique LIUPPA, Anglet, EA3000 – France
[2] LIUPPA – Université de Pau et des Pays de l'Adour : EA3000 – France

GoF's design patterns are popular structures designed for flexible and reusable object-oriented software. However, their energy impact is not well understood, with contradictory existing results. In this paper, we aim to answer the relation between design patterns and energy consumption through a multi-platform and multi-software stack empirical study, covering all design patterns, and multiple hardware, OS, languages and compilers. We found, with a high confidence level, that none of the GoF's design patterns has a significant impact on energy consumption. Therefore, they can be considered energy-neutral, allowing eco-friendly developers to continue using them. We argue that software energy efficiency should rather be considered within the business logic embedded by a design pattern, and ultimately raised to a higher perspective of software development.

*Intervenant

GT VL

Exploring Performance of Configurable Software Systems: the JHipster Case Study

Édouard Guégain * [1]

[1] Self-adaptation for distributed services and large software systems – Centre Inria de l'Université de Lille, Centre de Recherche en Informatique, Signal et Automatique de Lille - UMR 9189 – France

The performance of software systems remains a key concern in software engineering. Configurable software systems, with their numerous configurations, complicate the performance evaluation process. This paper investigates the impact of web stack configurations on performance, using the JHipster web stack generator as a case study. We analyze JHipster configurations to understand how component choices influence system performance and explore individual configuration options for their specific effects. Our study shows that correlations across performance indicators exist but are often weak, and different options affect performance unevenly, with some impacting one indicator minimally while significantly influencing another. We developed a performance model for JHipster to automate the identification of configurations optimized for specific metrics, identifying four configurations that outperform the current default. Overall, this study underscores JHipster's relevance as a use case for studying component-level variability in software systems and highlights the importance of selecting configurations based on performance indicators rather than preferred technologies.

*Intervenant

Does Functional Package Management Enable Reproducible Builds at Scale? Yes

Théo Zimmermann * [1]

[1] Laboratoire Traitement et Communication de lÍnformation – Institut Mines-Télécom [Paris], Télécom Paris – France

Reproducible Builds (R-B) guarantee that rebuilding a software package from source leads to bitwise identical artifacts. R-B is a promising approach to increase the integrity of the software supply chain, when installing open source software built by third parties. Unfortunately, despite success stories like high build reproducibility levels in Debian packages, uncertainty remains among field experts on the scalability of R-B to very large package repositories. In this work, we perform the first large-scale study of bitwise reproducibility, in the context of the Nix functional package manager, rebuilding 709 816 packages from historical snapshots of the nixpkgs repository, the largest cross-ecosystem open source software distribution, sampled in the period 2017–2023. We obtain very high bitwise reproducibility rates, between 69 and 91% with an upward trend, and even higher rebuildability rates, over 99%. We investigate unreproducibility causes, showing that about 15% of failures are due to embedded build dates. We release a novel dataset with all build statuses, logs, as well as full "diffoscopes": recursive diffs of where unreproducible build artifacts differ.

*Intervenant

Roseau: Fast, Accurate, Source-based API Breaking Change Analysis in Java

Corentin Latappy * [1]

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

Understanding API evolution and the introduction of breaking changes (BCs) in software libraries is essential for library maintainers to manage backward compatibility and for researchers to conduct empirical studies on software library evolution. In Java, tools such as Japicmp and RevApi are commonly used to detect BCs between library releases, but their reliance on binary JARs limits their applicability. This restriction hinders large-scale longitudinal studies of API evolution and fine-grained analyses such as commit-level BCs detection. In this paper, we introduce Roseau, a novel static analysis tool that constructs technology-agnostic API models from library code equipped with rich semantic analyses. API models can be analyzed to study API evolution and compared to identify BCs between any two versions of a library (releases, commits, branches, etc.). Unlike traditional approaches, Roseau can build API models from either source code or bytecode and is optimized for large-scale longitudinal analyses of library histories. We assess the accuracy, performance, and suitability of Roseau for longitudinal studies of API evolution, using Japicmp and RevApi as baselines. We extend and refine an established benchmark of BCs and show that Roseau achieves higher accuracy (F1 = 0.99) than Japicmp (F1 = 0.86) and RevApi (F1 = 0.91). We analyze 60 popular libraries from Maven Central and find that Roseau delivers excellent performance, detecting BCs between versions in under two seconds, including in libraries with hundreds of thousands of lines of code. We further illustrate the inadequacy of Japicmp and RevApi for longitudinal studies and the novel analysis capabilities offered by Roseau by tracking the evolution of Google's Guava API and the introduction of BCs over 14 years and 6,839 commits, reducing analysis times from a few days to a few minutes.

*Intervenant

Fingerprinting and Building Large Reproducible Datasets

Romain Lefeuvre * [1]

[1] Diversity-centric Software Engineering – Centre Inria de l'Université de Rennes, LANGAGE ET GÉNIE LOGICIEL – France

Obtaining a relevant dataset is central to conducting empirical studies in software engineering. However, in the context of mining software repositories, the lack of appropriate tooling for large scale mining tasks hinders the creation of new datasets. Moreover, limitations related to data sources that change over time (e.g., code bases) and the lack of documentation of extraction processes make it difficult to reproduce datasets over time. This threatens the quality and reproducibility of empirical studies. In this paper, we propose a tool-supported approach facilitating the creation of large tailored datasets while ensuring their reproducibility. We leveraged all the sources feeding the Software Heritage append-only archive which are accessible through a unified programming interface to outline a reproducible and generic extraction process. We propose a way to define a unique fingerprint to characterize a dataset which, when provided to the extraction process, ensures that the same dataset will be extracted. We demonstrate the feasibility of our approach by implementing a prototype. We show how it can help reduce the limitations researchers face when creating or reproducing datasets.

*Intervenant

GT GLSec

A Model-based Approach for Assessing the Security of Cyber-Physical Systems

Dominique Blouin *† [1]

[1] Laboratoire Traitement et Communication de l'Information (LTCI) – Institut Mines-Télécom [Paris], Télécom Paris – Télécom Paris 19 Place Marguerite Perey 91120 PALAISEAU, France

Cyber-Physical Systems (CPSs) complexity has been continuously increasing to support new life-impacting applications, such as Internet of Things (IoT) devices or Industrial Control Systems (ICSs). These characteristics introduce new critical security challenges to both industrial practitioners and academics. This work investigates how Model-Based System Engineering (MBSE) and attack graph approaches could be leveraged to model secure Cyber-Physical System solutions and identify high-impact attacks early in the system development life cycle. To achieve this, we propose a new framework that comprises (1) an easily adoptable modeling paradigm for Cyber-Physical System representation, (2) an attack-graph-based solution for Cyber-Physical System automatic quantitative security analysis, based on the MulVAL security tool, (3) a set of Model-To-Text (MTT) transformation rules to bridge the gap between SysML and MulVAL. We illustrated the validity of our proposed framework through an autonomous ventilation system example. A Denial of Service (DoS) attack targeting an industrial communication protocol was identified and displayed as attack graphs. In future work, we intend to connect the approach to dynamic security databases for automatic countermeasure selection.

*Intervenant

†Auteur correspondant: dominique.blouin@telecom-paris.fr

Gestion automatisée des consentements pour la protection des données personnelles

Selena Lamari *† [1]

[1] MaReL (LIRMM) – Laboratoire d'Informatique de Robotique et de Microélectronique de Montpellier – France

La gestion du consentement est d'une importance capitale lorsqu'il s'agit de traiter des données personnelles. Avec la montée des préoccupations en matière de protection de la vie privée et les réglementations strictes en matière de protection des données personnelles telles que le Règlement général sur la protection des données (RGPD), il devient obligatoire de s'assurer que ces données sont collectées et traitées de manière éthique et légale. Pour ce faire, les fournisseurs d'applications doivent traiter les consentements à la collecte et au traitement des données personnelles de manière transparente, explicite et simple. En outre, ils doivent également veiller à ce que ces consentements soient respectés en amont du cycle de vie de l'application. Notre travail consiste à présenter une approche pour gérer le consentement de l'utilisateur. Nous avons conçu une architecture et développé un protocole de contrôle d'accès comprenant l'authentification, la gestion des autorisations et la gestion des consentements externalisés, sur la base du modèle d'architecture ABAC (Attribute-Based Access Control) avec le schéma d'architecture side-car. L'architecture est basée sur le style d'architecture microservice. L'avantage de notre solution est qu'elle permet d'intégrer les fonctionnalités de contrôle d'accès et de gestion du consentement avec un minimum de modifications du code d'application du fournisseur. Notre solution est testée sur un exemple d'application qui gère des données personnelles. Cette expérimentation a montré que l'effort de développement pour intégrer le gestionnaire de consentement que nous proposons est léger et non intrusif pour le fournisseur de l'application.

*Intervenant

†Auteur correspondant: Selena.Lamari@lirmm.fr

Lutte Informatique d'Influence - Une approche conceptuelle pour la détection de Fakes News

Nicolas Belloir *† [1]

[1] IRISA (IRISA-SecReizh) – Institut de Recherche en Informatique et Systèmes Aléatoires – France

L'actualité internationale de plus en plus tendue, corrélée au pouvoir d'expression et de diffusion des réseaux sociaux notamment, forment un terreau fertile dans lequel la désinformation et l'influence peuvent largement se développer. Les Fakes News sont un des supports à cette désinformation. Elles sont largement répandues, créées parfois par de simples individus mais de plus en plus sous l'impulsion de groupes hostiles, que cela soit dans la sphère politique, culturelle ou économique.

La lutte informatique d'influence (L2I) est un des trois piliers, avec la lutte informatique défensive (LID) et la lutte informatique offensive (LIO), de l'arsenal Cyber dont s'est doté le gouvernement français face à la croissance des cybermenaces. La L2I nécessite des outils et des méthodes capables de relever plusieurs défis scientifiques. D'une part, la massification des Fake News va de paire avec des outils pour les contrer, basés sur des briques d'intelligence artificielle, de traitement automatique de langage naturel, d'analyse sémantique et statistique... Les travaux actuels, s'ils sont prometteurs, peinent à fournir une approche holistique. De plus, dans le domaine de la désinformation, il faut être capable de convaincre et d'expliquer. Or les techniques d'IA peinent à le faire, ce qui par ailleurs a ouvert un large champs scientifique dans le cadre de l'explicabilité. Enfin, les Fakes New sont souvent les composantes d'une attaque informationnelle plus complexe, qu'il faut être capable de détecter et d'analyser.

Dans ce contexte, nous proposons ici une approche basée sur l'utilisation de modèles conceptuels visant à structurer l'univers de la désinformation. Notre hypothèse est que, si une approche peut s'appuyer sur des modèles conceptuels pour déterminer et caractériser une FakeNews, alors cela permet d'expliquer le résultat donné. Nous présentons également nos premiers résultats visant à fournir une approche outillée capable, à partir d'une basses de données, construite sur notre modèle conceptuel de Fake News, d'aider un opérateur à détecter et/ou caractériser des attaques informationnelles. Notre approche met en avant l'apport de techniques issues du Génie Logiciel dans ce champ spécifique du domaine Cyber.

*Intervenant
†Auteur correspondant: nicolas.belloir@irisa.fr

GT IE + HIFI

Exigences de qualité extra-fonctionnelles des LLMs

Jean-Michel Bruel * [1]

[1] IRIT – Institut de recherche en informatique de Toulouse - IRIT – France

Cet exposé présentera les challenges et verrous liés à l'expression de propriétés pour les systèmes à base de LLMs, où les techniques classiques d'expression et de validation sont challengées par l'impossibilité de rendre complètement explicables (sans même parler de déterminisme) les algorithmes à la manœuvre. L'un des objectifs du projet ANR FATES-MLOps est de traiter les propriétés FATES (Fairness/Accountability/Transparency/Ethics/Security), c'est à dire être capable de déterminer à quel point les algorithmes de ML respectent (ou pas) ces propriétés. Dans ce cadre, l'expression, la caractérisation, de ces propriétés est une étape cruciale qui n'est pas sans relever de la mission impossible. Nous espérons susciter discussions et idées a autour de ce beau challenge.

*Intervenant

Leveraging Verification to Enhance Formal Explainable AI for Neural Networks

Ryma Boumazouza * [1], Mélanie Ducoffe [1]

[1] Airbus Defense and Space, Toulouse – Airbus Defense and Space, Toulouse – France

Dans le domaine de l'explicabilité des modèles, en particulier pour les réseaux de neurones, il est crucial de fournir des explications formelles garantissant à la fois la validité et la précision des résultats. Cependant, les algorithmes dans ce domaine présentent une limitation fondamentale : leur traitement séquentiel qui limite la parallélisation. Ce mode de fonctionnement, où les contraintes sont exclues une par une d'un ensemble minimal, devient particulièrement problématique dans le cadre des explications abductives lorsque le nombre de variables est très élevé. Pour des réseaux de neurones complexes, chaque appel visant à vérifier l'existence d'un exemple adversaire est coûteux en temps, et l'exécution séquentielle de ces appels crée un important goulot d'étranglement.

Pour pallier ces difficultés, notre contribution propose une approche innovante qui combine des garanties formelles avec l'efficacité pratique de l'intepretation abstraite et des attaques adversaires. Nous réemployons des outils de vérification de robustesse issus du machine learning pour les appliquer à l'explicabilité des réseaux de neurones. En tirant parti des approches récentes qui permettent de passer à l'échelle sur des architectures complexes, notre algorithme identifie les variables d'entrée responsables des défaillances de robustesse.

Cette méthode constitue une alternative robuste aux approches posthoc heuristiques telles que LIME, SHAP ou Anchor. Pour réduire le nombre d'appels séquentiels, nous privilégions un traitement parallèle d'un maximum de variables, afin de détecter à la fois des contreexemples pertinents et d'identifier le plus grand sous-ensemble de variables n'influençant pas la prédiction. Nos travaux en cours, actuellement non publiés, démontrent des résultats prometteurs sur des benchmarks d'images (MNIST) ainsi que dans un cas d'application aéronautique, avec des gains significatifs en temps de calcul et une réduction notable de la taille des explications (60% de réduction en taille d'explication et 80% en runtime). Ces résultats ouvrent des pistes intéressantes pour allier rigueur formelle et efficacité pratique dans l'explication des modèles de réseaux de neurones.

*Intervenant

Discussion

Sylvain Conchon * [1], Aurélie Hurault [2], Régine Laleau [3], Thomas Polacsek [4]

[1] Laboratoire Méthodes Formelles – Université Paris-Saclay, Centre National de la Recherche Scientifique – France
[2] Université de Toulouse, IRIT-ENSEEIHT – Université de Toulouse, IRIT-ENSEEIHT – France
[3] Laboratoire d'Algorithmique Complexité et Logique – Université Paris-Est Créteil Val-de-Marne - Paris 12 – France
[4] ONERA, Université de Toulouse [Toulouse] – ONERA – France

Discussion sur l'utilisation des méthodes formelles pour l'ingéniérie des exigences.

*Intervenant

GT CLAP + YODA

Simulations et compilation vérifiée

Nicolas Chappe * [1]

[1] Verimag – Verimag, CNRS, Grenoble – France

Les preuves de correction de compilateur reposent sur des résultats d'inclusion de comportements, qui prennent en général la forme d'une simulation entre des systèmes de transition étiquetés (LTS).
Dans cet exposé nous proposons une nouvelle notion de simulation, avec une définition relativement simple proche de la simulation faible sensible à la divergence classique, mais qui admet des principes de raisonnement puissants grâce à l'utilisation de la coinduction up-to. Cette simulation est formalisée en Rocq dans un cadre abstrait de LTS, ce qui permet de l'instancier sur divers projets en lien avec la compilation vérifiée, notamment sur CompCert ou encore sur des structure de la famille des arbres d'interaction, par exemple les Choice Trees.

*Intervenant

Vérification formelle de propriétés interactives: retours d'expérience

Cécile Marcon * [1]

[1] Fédération ENAC ISAE-SUPAERO ONERA – Institut Supérieur de l'Aéronautique et de l'Espace, ONERA, Ecole Nationale de l'Aviation Civile – France

Nous nous intéressons à la vérification formelle de systèmes interactifs, en particulier à la correction du langage Smala, un UIDL (User Interface Description Language). Pour cela, nous souhaitons vérifier des propriétés interactives (comme la visibilité, la perceptibilité, ou la possibilité d'annuler une action) sur un UIDL formel: Biguil, un langage basé sur les bigraphes.
À partir d'une approche centrée utilisateur, nous avons établi une liste de propriétés que nous souhaiterions vérifier, ainsi qu'un langage d'annotations, RAIS, permettant d'exprimer ces propriétés directement dans le code Smala.

Nous avons entamé une chaîne de compilation qui se termine par la vérification par model checking d'une propriété interactive.

*Intervenant

GT MTV2 - AFADL

Apprentissage et test pour les machines à états finis temporisées avec délais de sortie

Evgenii Vinarskii * [1], Natalia Kushik [1], Djamal Zeghlache [1]

[1] Télécom SudParis – Institut Polytechnique de Paris – France

Le travail est consacré à l'analyse et à la synthèse des machines à états finis temporisées avec délais de sortie (angl. Timed Finite State Machines ou TFSMs with output delays). Nous résumons brièvement nos résultats publiés récemment dans le domaine de l'apprentissage de TFSM et la dérivation de la suite de tests à partir de la machine dérivée. Pour dériver la suite de tests, nous utilisons la stratégie du "timed transition tour (TTT)". La nouvelle contribution à l'AFADL consiste en l'évaluation expérimentale de la corrélation entre les stratégies de sélection d'horodatage (angl. timestamp) et le modèle de fautes / taux de couverture du TTT. Nous réalisons des expériences pour évaluer l'efficacité des stratégies de tests, avec des implémentations mutées du logiciel Ansible qui est un outil de gestion de configuration, utilisé dans le "cloud computing".

*Intervenant

Experimental Evaluation of LLMs for Test Generation

Tiago Costa [1], Ahmad Ghandour [1], Mariia Soltys [1], Nikita Volosnikov [1], Yves Ledru * [1], Nicolas Hili [1]

[1] Laboratoire d'Informatique de Grenoble – Institut National de Recherche en Informatique et en Automatique, Centre National de la Recherche Scientifique, Université Grenoble Alpes, Institut Polytechnique de Grenoble - Grenoble Institute of Technology – France

With the increased value of information, developing and testing robust software applications within shorter production cycles has become increasingly challenging. The recent outbreak in Research and development of Large Language Models (LLMs) presents an opportunity to rethink software development by automating as many repetitive tasks as possible. In particular, leveraging LLMs for both black-box and white-box testing is of significant interest.

This paper provides an experience report assessing the quality of test cases generated by LLMs on the same java function, using a variety of prompts to produce black-box and white box test suites. It evaluates the quality of the resulting test suites using the following criteria : absence of build errors, correctness of the test oracle, capability to find failures and code coverage.

*Intervenant

Projet ASMA VéDySec : Vérification Dynamique de Propriétés de Sécurité

Nikolai Kosmatov * [1], Frédéric Loulergue [2], Julien Signoles [3]

[1] Thales Research and Technology, cortAIx Labs – THALES – France
[2] Laboratoire d'Informatique Fondamentale d'Orléans – Université d'Orléans, Institut National des Sciences Appliquées - Centre Val de Loire, Université d'Orléans : EA4022 – France
[3] CEA LIST – Université Paris Saclay, Université Paris-Saclay – France

Cette soumission présente le projet Astrid Maturation VéDySec (Vérification Dynamique de Propriétés de Sécurité) qui a été soumis et accepté en 2024, et démarre début 2025, avec une durée de 24 mois. Il porte sur les techniques de spécification et vérification à l'exécution de propriétés de sécurité et vise le passage du TRL 4 au TRL 6. Le consortium inclut Thales TRT (coordinateur : Nikolai Kosmatov), le LIFO (responsable scientifique : Frédéric Loulergue) et le CEA List (responsable scientifique : Julien Signoles). Ce projet fait suite aux thèses CEA-DGA de Dara Ly et de Virgile Robles.

*Intervenant

Projet ANR RAPID VVaMIA - Vérification et Validation des systèmes embarqués augmentés par les Modèles et l'IA générative

Frédéric Dadeau [1], Fabrice Bouquet [1], Eléa Jacquin [1], Dorine Tabary * [1], Louis Lefevre [1], Bruno Legeard [2], Antoine Chevrot [2], Arnaud Bouzy [2], Bruno Besace [3], Fabien Lamontre [3]

[1] FEMTO-ST Institute – Université Marie et Louis Pasteur – France
[2] Smartesting Solutions et Services – Smartesting Solutions et Services – France
[3] Thales LAS France – THALES – France

Ce papier présente les motivations, objectifs et travaux prévus dans le projet ANR RAPID VVaMIA, porté par l'entreprise Smartesting Solution & Services. Le consortium inclut Thales LAS, ainsi que FEMTO-ST. Ce projet s'intéresse au couplage entre l'IA générative et les modèles, dans le contexte de la vérification et la validation de systèmes embarqués en utilisant des tests logiciels. Ce projet vise à la production d'assistants IA facilitant les tâches de conception et de maintenance des tests logiciels.

*Intervenant

GT LVP - AFADL

BICOQ : une formalisation des bigraphes dans Coq

Cécile Marcon * [1], Cyril Allignol [2], Célia Picard [2], Blair Archibald [3], Michele Sevegnani [4], Xavier Thirioux [5]

[1] ISAE-SUPAERO – ISAE-SUPAERO – France
[2] ENAC – ENAC – France
[3] University of Glasgow – Royaume-Uni
[4] University of Glasgow – Royaume-Uni
[5] ISAE-SUPAERO – ISAE-SUPAERO – France

Dans l'article "BiCoq : Bigraphs Formalisation with Coq", nous présentons une librairie Coq (open-source) sur les bigraphes. Nous y détaillons le type choisi pour représenter un bigraphe, les définitions d'équivalence, ainsi que des opérateurs de composition, produit tensoriel, produit parallèle et des opérateurs dérivés que nous avons implémentés. Nous donnons les théorèmes -énoncés par Milner- que nous avons prouvés qui nous permettent de nous assurer de la correction de notre implémentation. Notre objectif étant de disposer d'une implémentation complète et vérifiée de la théorie des bigraphes, cette première étape est un jalon important.

*Intervenant

Une sémantique mécanisée d'un langage FRP avec effets

Jordan Ischard * [1], Frederic Dabrowski [1], Jules Chouquet [1], Frédéric Loulergue [1]

[1] LIFO – Université d'Orléans – France

La programmation fonctionnelle réactive (FRP) est un paradigme de programmation fonctionnelle conçu pour simplifier la programmation de systèmes interagissant continuellement avec leur environnement. Yampa, une bibliothèque Haskell, permet de construire des fonctions de signal qui traitent un flux d'entrée de manière synchrone afin de produire un flux de sortie. Bien que cette bibliothèque facilite la production de programmes concis et robuste, la gestion des entrées et sorties reste fastidieuse. Pour résoudre ce problème, la bibliothèque Wormholes étend Yampa avec des constructions rendant possible des interactions avec l'environnement en dehors du flux d'entrée dans un style impératif. Ce résumé présente notre article de mécanisation d'une légère variante du langage Wormholes qui corrige les problèmes de la formalisation initiale.

*Intervenant

Prouvez vos coloriages: vérification formelle du coloriage de cache de l'hyperviseur Bao

Axel Ferréol [1], Laurent Corbin [1], Nikolai Kosmatov * [1]

[1] cortAIx Labs – Thales (France) – France

Cette soumission est un résumé long d'un papier accepté à FASE 2025 (attaché à la fin du résumé). L'objectif de ce travail est la vérification formelle du mécanisme de coloriage de cache implémenté dans Bao, un hyperviseur statique open-source utilisé dans les systèmes embarqués. Bien qu'il propose une implémentation élégante et optimisée, son code est complexe à vérifier formellement en raison de la logique non triviale, des opérations au niveau des bits, des calculs arithmétiques complexes, des niveaux multiples de boucles imbriquées et des listes chaînées qu'il utilise. Au cours de cette étude de cas, nous avons identifié deux bugs subtils dans l'implémentation existante, cassant les garanties attendues, et avons proposé des correctifs. Nous avons écrit une spécification formelle pour les fonctions clés du mécanisme et vérifié leur version corrigée dans la plateforme de vérification Frama-C.

*Intervenant

Langage Chips : Modélisation et contrôle de systèmes distribués à base de composants

Anna Gallone * [1]

[1] FEMTO-ST – Université Marie et Louis Pasteur, CNRS, institut FEMTO-ST (UMR 6174), F-25000 Besançon, France – France

Ce document présente Chips, un langage de programmation dédié à la modélisa- tion de systèmes à base de composants. Son objectif est de permettre la modélisation d'architectures logicielles auxquels une forme de contrôle en temps réel serait appli- quée. Pour cela, Chips intègre des constructions syntaxiques facilitant l'expression des formalismes mathématiques issus de la théorie du contrôle. De plus, pour être appliqué à des systèmes logiciels distribués, le langage intègre des concepts de programmation agrégée telle la généralisation de l'opération de broadcast ou d'accumulation. En compilant Chips vers le langage BIP, il sera possible de vérifier par preuve ou model- checking des propriétés de résilience des systèmes conçus sur des automates équivalents aux programmes initiaux.

*Intervenant

AFADL

Découverte de processus probabiliste avec des arbres de processus stochastiques

Pierre Cry * [1]

[1] Université Paris-Saclay, CentraleSupélec, Laboratoire MICS – Centrale Supélec – France

La découverte de processus vise à extraire automatiquement un modèle formel capable de reproduire les comportements observés d'un système à partir de journaux d'événements. Ces journaux décrivent des séquences d'événements qui se produisent avec une certaine fréquence, reflétant ainsi la nature stochastique du processus sous-jacent. Dans ce travail, nous adoptons une extension stochastique du formalisme des arbres de processus comme modèle cible pour la découverte de processus. Nous montrons que ces modèles offrent des avantages par rapport aux réseaux de Petri stochastiques, en particulier pour l'identification des paramètres influençant la probabilité des comportements observés. Ce document est un résumé long du papier " Probabilistic Process Discovery with Stochastic Process Trees ", accepté à EAI VALUETOOLS 2024, co-écrit avec András Horváth et Paolo Ballarini.

*Intervenant

Tail Modulo Async-Await

Emma Nardino * [1], Ludovic Henrio [2], Gabriel Radanne [3], Yannick Zakowski [3]

[1] ENS Lyon, Univ Lyon, UCBL, CNRS, Inria, LIP, France – École Normale Supérieure - Lyon – France
[2] CNRS, Univ Lyon, ENS Lyon, UCBL, Inria, LIP, France – Centre National de la Recherche Scientifique - CNRS – France
[3] Inria, Univ Lyon, ENS Lyon, UCBL, CNRS, LIP, France – INRIA – France

This article extends tail-call optimisation by applying it to asynchronous calls. We first introduce Tail-Modulo-Await, a novel code transformation for asynchronous tail recursive functions that prevents the creation of unnecessary tasks. We then show how to combine Tail-Modulo-Await with the existing Tail-Modulo-Cons optimisation; we obtain an optimisation able to turn a recursive function with multiple tail calls under constructors into a parallel version of the function, also optimised in space. We formalise both optimisations over representative calculi, and prove them correct through backward simulations. Finally, we provide a proof-of-concept implementation as an OCaml syntax extension and evaluate it experimentally, showing our approach optimises both memory and execution time.

*Intervenant

Model Checking de LTL sur Traces Finies et Infinies avec Domaines Concrets

Julien Brunel * [1], David Doose [1]

[1] DTIS, ONERA, Université de Toulouse – ONERA – France

Il existe différentes sémantiques pour la logique temporelle linéaire (LTL) concernant la prise en compte de traces finies. Bien que plusieurs d'entre elles puissent s'avérer utiles en fonction du contexte de vérification, aucun cadre de vérification ne permet de gérer leur diversité de manière simple. Une autre limitation des outils de vérification de LTL actuels est leur incapacité à traiter les domaines concrets (entiers bornés et infinis, nombres réels, etc.). Nous présentons une approche permettant de faire du model checking de LTL sur des traces finies et infinies avec des domaines concrets. Notre méthode repose sur un solveur SMT et sur le model checking borné (BMC). Nous présentons également quelques expérimentations et comparons notre outil avec NuSMV et nuXmv.

*Intervenant

VASSAL Project: Verification and Analysis for Safety and Security of Applications in Life

Julien Signoles * [1], Milan Ceska [2], Florian Zuleger [3], Tomáš Kratochvíla [4]

[1] Université Paris-Saclay, CEA, List – CEA-LIST – France
[2] Brno University of Technology – République tchèque
[3] Technische Universität Wien – Autriche
[4] Honeywell International – États-Unis

This submission aims at presenting the EU Twinning project VASSAL (Verification and Analysis for Safety and Security of Applications in Life) that was started in June 2024 and runs for 36 months. The beneficiary partners are Brno University of Technology (BUT) in the Czech Republic, Vienna University of Technology (TUW) in Austria, and CEA List in France (CEA). Honeywell International (HON) is an associate partner. The project coordinator is Milan Češka from BUT.

*Intervenant

Approche neuro-symbolique pour générer un référentiel de test

Eléa Jacquin * [1]

[1] Université Marie et Louis Pasteur - FEMTO-ST – Université Marie et Louis Pasteur, CNRS, institut FEMTO-ST (UMR 6174), F-25000 Besançon, France – France

Ce papier présente l'application de méthodes neuro-symboliques pour automatiser la génération de tests logiciels en combinant les capacités d'interprétation sémantique des Large Language Models (LLMs) avec la rigueur formelle des méthodes symboliques. Ce travail sera appliqué dans le cadre du projet ANR VVaMIA qui a démarré début 2025.

*Intervenant

Transformations automatisés de preuves Coq

Alexandre Jean * [1], Nicolas Magaud [1]

[1] ICube UMR 7357 CNRS - Université de Strasbourg – ICube UMR 7357 CNRS - Université de Strasbourg – France

Les assistants de preuves interactifs tel que Coq permettent à leurs utilisateurs d'écrire et de vérifier des preuves sur des sujets variés. Dans cette thèse, nous explorons le cycle de vie d'une preuve Coq après son écriture, notamment la phase de maintenance mais aussi les différentes réécritures possibles de cette preuve. Dans cet objectif, nous présentons dans ce document nos travaux actuels consistant en une bibliothèque de réécriture et d'exécution d'AST Coq ainsi qu'un exemple de transformation réalisable à l'aide de cette bibliothèque.

*Intervenant

Langage de spécification de contexte pour vérifier formellement des propriétés de consentement sur des modèles et du code

Myriam Clouet * [1], Thibaud Antignac [2], Mathilde Arnaud [3], Julien Signoles [3]

[1] Université d'Orléans, LIFO – Université d'Orléans – France
[2] CNIL – CNIL – France
[3] Université Paris-Saclay, CEA, List – Université Paris-Saclay, CEA, LIST, Université Paris-Saclay, CEA List – France

Cette soumission présente l'article Context Specification Language for Formally Verifying Consent Properties on Models and Code qui a été publié à la conférence Tests and Proofs en 2023 (5). Il porte sur la définition d'un langage formel, la formalisation de propriétés liées au consentement et à l'implémentation d'un outil de vérification formelle. Cet outil peut être utilisé soit pour vérifier une modélisation d'un système, soit pour vérifier le code correspondant à son implémentation.

*Intervenant

Project & Conquer : Élimination de quantificateurs pour la vérification de problèmes d'accessibilité dans les réseaux de Petri

Nicolas Amat * [1], Silvano Dal Zilio [2], Didier Le Botlan [2]

[1] ONERA – ONERA, ONERA – France
[2] LAAS-CNRS – CNRS : UPR0081, CNRS, CNRS – France

Dans cet article nous présentons une méthode pour accélérer la vérification de propriétés d'accessibilité dans les réseaux de Petri qui tire parti de réductions structurelles et qui peut être utilisée, de manière transparente, comme étape de prétraitement de model-checkers existants. Notre approche est basée sur une nouvelle procédure qui permet de projeter une propriété, concernant un réseau de Petri, en une formule équivalente qui se réfère uniquement à une version réduite de ce réseau. Notre projection est définie comme une procédure d'élimination de quantificateurs pour l'arithmétique linéaire entière (ou arithmétique de Presburger), adaptée au type spécifique de contraintes que nous traitons. Les résultats expérimentaux montrent que notre approche fonctionne bien en pratique et qu'elle peut être utile même lorsqu'il n'y a qu'une quantité limitée de réductions.

*Intervenant

Posters et démos

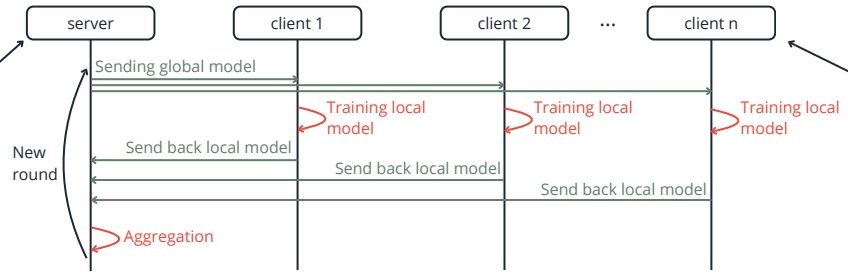
Approaches to strengthen embedded AI against attacks disrupting federated learning

Camille Molinier, Paul Temple, Olivier Zendra, Olivier Barais (Univ Rennes, CNRS, Inria, Irisa)

More information

1 FEDERATED LEARNING[1]

The server:
- Holds and trains the global model
- Combines the local models
- Never sees data



The Clients:
- Hold and train local models
- Collect local data
- Are heterogeneous (data, ...)

Data never leave clients but the model do

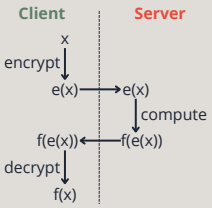
- Machine learning model behavior relies on data
- In production, data can evolve due to:
  - Natural data drift
  - New class appearance
  - etc

Federated learning integrity can be compromised in many ways:
- Data poisoning
- Backdoors introduction in the model
- Membership inference

Federated learning is vulnerable[2], it needs strengthening

2 CURRENT DEFENSE TECHNIQUES[2]

Homomorphic encryption

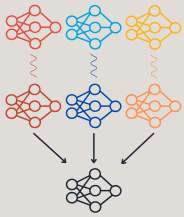


- Clients can have their encryption function
- Functions need to be additive

- Resource consuming
- Not suitable in distributed environments

REJECTED

Differential privacy



- Privacy method
- Adds noise to the local gradient
- Complicates inference attack

- Not secure enough[3]
- Mitigates attacks, doesn't defend
- Can make attacks easier

REJECTED

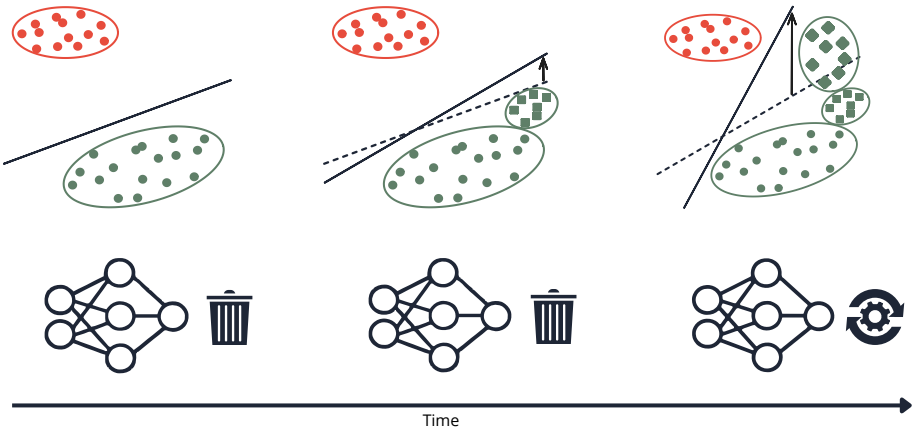
Secure aggregation protocol

- Federated averaging can be abused
- Security-aware aggregation methods (FedSec+, robust aggregation, etc)

- Can be hardware intensive
- Harms convergence
- Not widely adopted

REJECTED

3 OUR APPROACH



Federated learning provides many variation points

- Data distribution varies among clients
- Local and global models evolve over time
- Previous global models are dropped after aggregation
- Machine learning variation points:
  - Random model weight initialization
  - Data train / test split
  - Batch shuffling

Techniques to manage variability exist

- Federated learning offers variability
- Techniques exist in software engineering to manage (software) variability
- In the context of federated learning, we want to leverage these software variability management techniques

Thesis research questions :

RQ1: Can the history of global models be used to detect/prevent data drift?

RQ2: Can software testing methods be leveraged to develop defense mechanisms for federated learning?

5 ACKNOWLEDGMENT

This work as been funded by DGA and CMA CyberSkills4All

Université de Rennes · IRISA · cnrs · Inria · DiverSE

REFERENCES

1. H. B. Mc Mahan, E. Moore, D. Ramage and B. A. y Arcas, "Federated learning of deep networks using model averaging,"CoRR, vol. abs/1602.05629, 2016. http://arxiv.org/abs/1602.05629
2. J. Hasan, "Security and privacy issues of federated learning," 2023. https://arxiv.org/abs/2307.12181
3. J. Wang, R. Schuster, I. Shumailov, D. Lie, and N. Papernot, "Indifferential privacy, there is truth: On vote leakage in ensemble private learning," 2022. https://arxiv.org/abs/2209.10732

Détection et Réparation d'Anomalies en Arithmétique à Virgule Flottante par Analyse Statique

Julien Bortolussi[1] Dorra Ben Khalifa[1] Pierre-Loïc Garoche[1]

[1]Fédération ENAC ISAE-SUPAERO ONERA, Université de Toulouse, France

Contexte

L'arithmétique à virgule flottante est l'approximation la plus courante de l'arithmétique réelle [1].

(a) Ariane 5 (2016)

(b) Supercalculateur de météo France

Utilisation des nombres flottants

Cette méthode a cependant un défaut, toutes les opérations sont susceptibles de subir une erreur d'arrondi. Bien que ces erreurs soient relativement faibles, elles peuvent être amplifiées ou conduire à des comportements contre-intuitifs. C'est ce que l'on appelle des anomalies flottantes.

L'objectif de cette thèse est de proposer une méthode basée sur l'analyse statique pour détecter les anomalies de l'arithmétique flottante et proposer des réparations.

Les Bases de L'Arithmétique Flottante

Un nombre flottant $\tilde{x}$ est représenté par :

$$\tilde{x} = b_0.b_1...b_{p-1} \times \beta^e \quad (1)$$

Où β est la base (généralement 2), $b_0.b_1...b_{p-1}$ la **mantisse**, e l'**exposant** et p la **précision**.

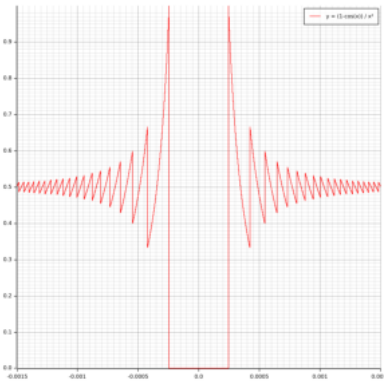
Le Standard **IEEE754** [1] définit pour les nombres flottants:

- Des **modes d'arrondi** ($\uparrow +\infty$, $\uparrow -\infty$, $\uparrow 0$, *round to nearest*).
- Des **formats** (32 bits, 64 bits, 128 bits...)
- Des opérateurs élémentaires qui doivent être exactement arrondis (évalués exactement puis arrondis)

L'Arithmétique flottante présente plusieurs anomalies [3]

Overflow	Underflow	NaN	Absorption	Cancellation
$+\infty$	~ 0	$\sqrt{-1}$	$1+\sim 0$	$1-cos(\sim 0)$

Voici un exemple de cancellation issue de [4]



Evaluation de $f(x) = \frac{1-cos(x)}{x^2}$ sur 32 bits (p=24)

Objectfs

- Générer des contraintes décrivant la propagation des erreurs dans les calculs pour détecter les anomalies de l'arithmétique flottante.
- Proposer des méthodes pour automatiquement corriger ces anomalies.

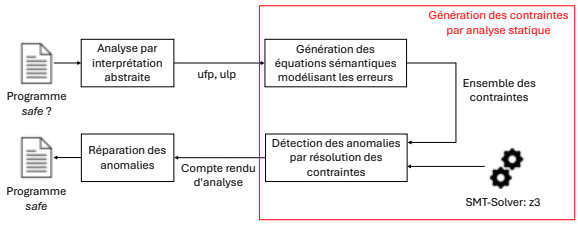
Génération de Contraintes par Analyse Statique

Notre approche se base sur l'approximation des nombres et de leur erreur par leur ordre de grandeur. Pour cela, nous utilisons les notions d'ufp, ulp et nsb [2].

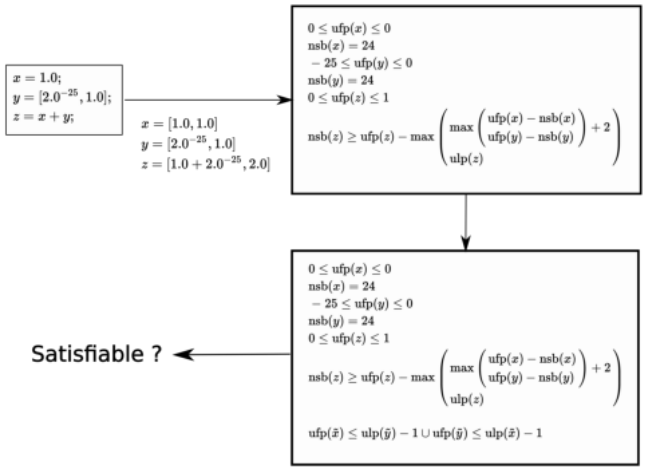
Notons $x \in \mathbb{R}$, $\tilde{x}$ sa représentation flottante et $\epsilon_x = x - \tilde{x}$ l'erreur sur $\tilde{x}$.

- **ufp** (Unit in the First Place): $\text{ufp}(\tilde{x}) = min\{i \in \mathbb{N} | 2^{i+1} > |\tilde{x}|\}$
- **ulp** (Unit in the Last Place): $\text{ulp}(\tilde{x}) = \text{ufp}(\tilde{x}) - p + 1$
- **nsb** (Number of Significant Bits): $\text{nsb}(\tilde{x}) = \text{ufp}(\tilde{x}) - \text{ufp}(\epsilon_x)$

Notre outil présente la structure suivante:



Une première phase par interprétation abstraite permet d'obtenir les ufp qui sont ensuite utilisées pour générer les contraintes portant sur les nsb.



Pour chaque anomalie possible, une contrainte est temporairement ajoutée au système. S'il devient insatisfiable alors l'anomalie ne peut pas se produire.

Perspectives

- Enrichir le langage (boucles, conditions, fonctions mathématiques...)
- Expérimenter de nouvelles méthodes de détection (interprétation abstraite)
- Proposer automatiquement des réparations pour les anomalies détectées
  - par transformation du programme [6]
  - par réglage de précision [5]
- Exercer l'outil sur un ensemble de benchmark (**FPBench**)

Références

[1] IEEE standard for floating-point arithmetic. *IEEE Std 754-2008*.

[2] D. Ben Khalifa. *Fast and efficient bit-level precision tuning*. Theses, Université de Perpignan, Nov. 2021.

[3] I. Laguna, T. Tirpankar, X. Li, and G. Gopalakrishnan. Fpchecker: Floating-point exception detection tool and benchmark for parallel and distributed hpc. In *2022 IEEE International Symposium on Workload Characterization (IISWC)*, pages 39–50. IEEE, 2022.

[4] M. O. Lam, J. K. Hollingsworth, and G. Stewart. Dynamic floating-point cancellation detection. *Parallel Computing*, 39(3):146–155, 2013.

[5] M. Martel. Floating-point format inference in mixed-precision. In *NASA Formal Methods Symposium*, pages 230–246. Springer, 2017.

[6] P. Panchekha, A. Sanchez-Stern, J. R. Wilcox, and Z. Tatlock. Automatically improving accuracy for floating point expressions. *SIGPLAN Not.*, 50(6):1–11, June 2015.

Intégration d'Exigences Exécutables dans le Prototypage

Yann Le Goff [1 2], Éric Le Pors [2], Pierre Laborde [2], Mickaël Kerboeuf [1], Alain Plantec [1]

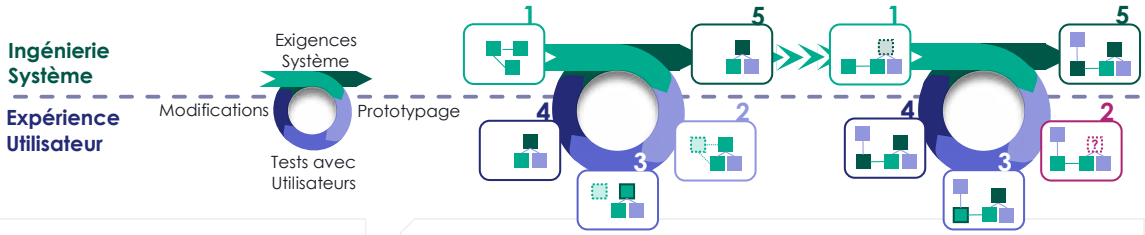
[1] Lab-STICC, CNRS, UMR 6285, UBO, 29200 Brest, France
[2] Thales DMS, 29200 Brest, France
[1]{firstname.name}@univ-brest.fr, [2] {firstname.name}@thalesgroup.com

> Contexte

- Le processus de développement applicatif chez Thales DMS implique une collaboration interdisciplinaire entre l'Ingénierie Système, l'UX Design et l'Ingénierie Logicielle, chacune disposant d'une vision partielle et spécialisée du système. L'Ingénierie Système adopte une vue macro, tandis que des disciplines comme l'UX se concentrent sur des sous-systèmes tels que les IHM. Les interdépendances entre ces disciplines rendent critiques les décisions locales, qui peuvent impacter d'autres domaines d'ingénierie. Or, l'absence d'outils d'analyse d'impact des exigences limite la maîtrise des effets de propagation des modifications.

> Problématique

- **Eviter la régression sur les exigences.**

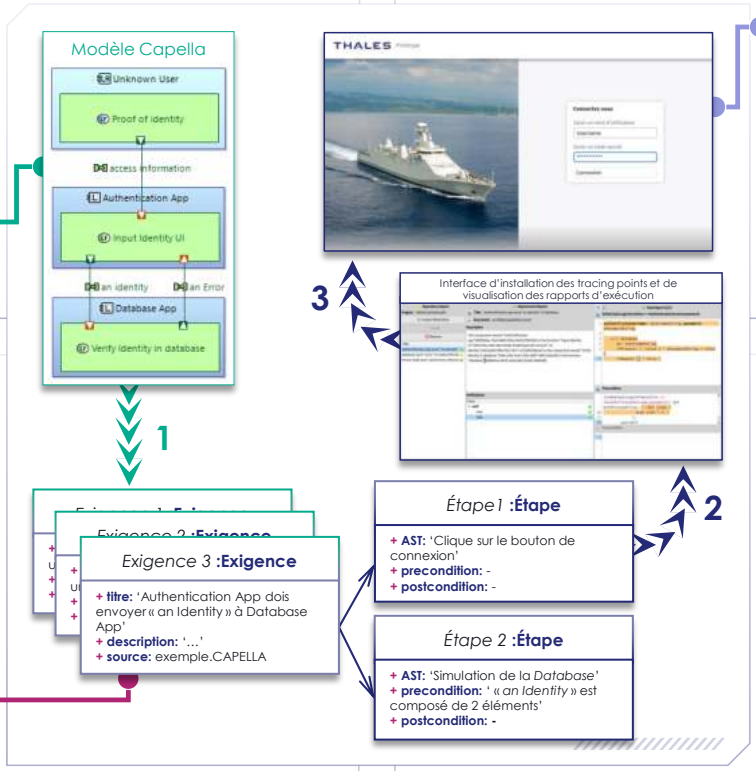


> Ingénierie Système

- **Objectif -** Approche interdisciplinaire qui gouverne l'ensemble de l'effort technique requis pour transformer une exigence en solution système.
- **Méthode Arcadia pour créer des modèles :**
  - Quels sont les besoins des utilisateurs du système ?
  - Qu'est-ce que le système doit accomplir pour les utilisateurs ?
  - Comment le système va-t-il répondre aux besoins ?
  - Comment va-t-il être construit, le système ?

> Expérience Utilisateur

- **Objectif -** Perceptions et réponses d'une personne résultant de l'utilisation ou de l'utilisation anticipée d'un produit, d'un système ou d'un service.
- **Création d'un Prototype**
  - Un prototype est utilisé pour obtenir des feedback des utilisateurs (au cours de GT ou d'évaluation avec les utilisateurs) et améliorer et spécifier une IHM complexe, pour étudier la faisabilité ou identifier les exigences.
    - Définition ISO 24765



> Proposition

- Création d'un méta-modèle d'exigences exécutables.
  - Titre, description, document d'origine.
  - Annotation du code appellé « Tracing Point » avec précondition et/ou postcondition.
- Transformation automatique des modèles Capella en exigences exécutable.
  - Transformation des modèles Capella vers un modèle Capella simplifié.
  - Sélection des exigences Capella liées à l'IHM à la main.
  - Transformation des exigences Capella en exigences exécutables.
  - Ajout à la main d'étapes de vérification associées à des « tracing points » .
- Les tracing points sont « exécutés » en même temps que le code associé, les précondition et les postconditions sont évaluées à la volée et un rapport d'exécution est produit pour chaque étape.

> Validation

- Implémentation du méta-modèle d'exigences exécutables dans l'environnement de développement Pharo.
- Extension du système de « Debug Point » (breakpoints de Pharo) pour implémenter les tracing points.
- Création d'un prototype d'authentification dans le langage Pharo.
- Création des exigences associées **(1)**.
  - Modèles Capella créer avec le logiciel Capella.
  - Transformation de ce modèle dans Pharo pour créer des exigences exécutables.
  - Enrichissement des exigences avec des étapes de vérifications (cible un nœud de l'AST du code, ajout des précondition et postcondition).
- Installation des tracing points à partir des étapes des exigences **(2)**.
- Exécution du prototype et de ses exigences associées **(3)**.

Journées Nationales du GDR GPL 2025 – Pau

Copyrigth Photo : Thales Netherland BV

The use of Digital Twins to Bolster the Resilience of IoT System-of-Systems

Laboratoire Décision et Information Pour les Systèmes de Production

POLYTECHNIQUE MONTRÉAL TECHNOLOGICAL UNIVERSITY

Meriem SMATI

meriem.smati@insa-lyon.fr
meriem.smati@polymtl.ca

10/2023 – 10/2026

Supervisors

- Jannik Laval
- Vincent Cheutet
- Christophe Danjou

Cotutelle INSA Lyon & Polytechnique Montréal

Context

IoT-based SoS are complex and vulnerable to disruptions due to their distributed and autonomous nature. DTs offer promising tools to improve resilience through real-time monitoring, simulation, and adaptation. However, their integration into IoT SoS raises scientific challenges related to modeling, synchronization, and system interoperability.

Research Questions

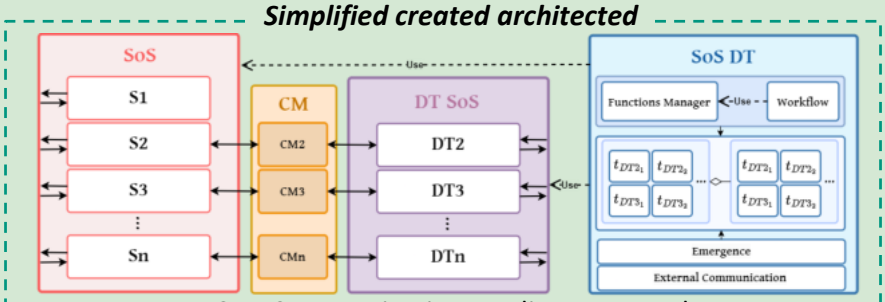
- How can a SoS DT be modeled to reflect the complexity of IoT SoS and support resilience?
- What resilience properties can be effectively enhanced using SoS DTs?
- What are the key scientific and technical barriers to implementing SoS DTs at scale?

Keywords: Digital Twins, Systems-of-Systems, Resilience, Internet-of-Things, Model-Driven Engineering

Methodology

Proposed SoS DT Architecture [1]

Simplified created architected



CM: Communication Medium t: task

Study to Conduct

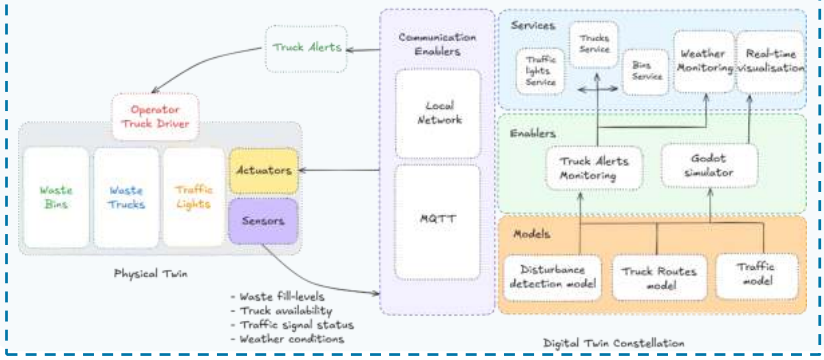
Develop & compare models :

- System of DTs
- SoS DT

Use Case: Smart Waste Collection SoS

General Inner Architecture

Simplified collaboration between the Bin DT and Trucks and Lights DS

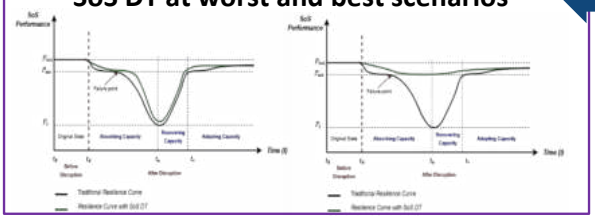


Results and Anticipations

- Bins DT
- Trucks DS
- Lights DS

WasteTwin Demo

SoS DT at worst and best scenarios



Smati, M., Cheutet, V., Danjou, C., & Laval, J. (2025, February). Digital Twin System of Systems: A Layered Architecture Proposal. In *13th International Conference on Model-Based Software and Systems Engineering (MODELSWARD 2025).*

2025

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES LYON · Université Claude Bernard Lyon 1 · Université Lumière Lyon 2 · Université Jean Monnet Saint-Étienne · Université de Lyon

Diffing build logs with CiDiff

Nicolas Hubner – Jean-Rémy Falleri, Raluca Uricaru

PROGRESS-Univ. Bordeaux, LaBRI, France

LaBRI

Introduction

```
name: Build project
on: [push]
jobs:
  build-project:
    runs-on: ubuntu-latest
      steps:
        - uses: actions/checkout@v4
        - uses: actions/setup-java@v4
          with:
            java-version: '17'
            distribution: 'temurin'
        - name: Setup Gradle
          uses: gradle/actions/setup-gradle@v4
        - name: Build with Gradle
          run: ./gradlew build
        - name: Upload build artifacts
          uses: actions/upload-artifact@v4
          with:
            name: Package
            path: build/libs
```

Setup java & gradle

Compilation

Upload results

↓

- Continuous Integration (CI) is widely used by developers to ensure the reliability of their software.
- However, when the CI fails, the result never explains *why* there is a problem. A developer needs to read the log to understand the cause of the error
- But the produced logs are lengthy and their exhaustive analysis is tedious, the vocabulary used to signal a problem is not consistent, some warnings and errors do not always trigger a failure, and a cascade of errors can span a large portion of the log.

Detection of the error using the diff between two logs

```
 1 Downloading: com.fasterxml.jackson:core:2.9.0
 2 Warning jackson-core has 2 vulnerabilities
-3 Downloading: com.github.gumtreediff:core:2.1.0
 4 Downloading: ch.qos.logback:logback-classic:1.1

 5 Downloading: org.scala-lang:scala-library:2.11
-6 Compiling package core: success
-7 Total time: 5.170 s
-8 Final Memory: 19M/176M
```

(a) Passing log

```
 1 Downloading: com.fasterxml.jackson:core:2.9.0
 2 Warning jackson-core has 2 vulnerabilities

 3 Downloading: ch.qos.logback:logback-classic:1.1
+4 Downloading: com.github.gumtreediff:core:2.1.0
 5 Downloading: org.scala-lang:scala-library:2.11
+6 Compiling package core: failure
+7 core/foo.java: unable to parse
+8 Total time: 5.361 s
+9 Final Memory: 19M/179M
+10 Error compiling project
```

(b) Failing log

A diff produced by our algorithm

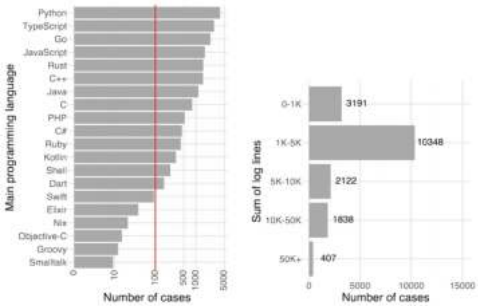
```
1 Downloading: com.fasterxml.jackson:core:2.9.0
2 Warning jackson-core has 2 vulnerabilities
3 Downloading: com.github.gumtreediff:core:2.1.0
4 Downloading: ch.qos.logback:logback-classic:1.1

5 Downloading: org.scala-lang:scala-library:2.11
6 Compiling package core: success

7 Total time: 5.170 s
8 Final Memory: 19M/176M
```

(a) Passing log

```
 1 Downloading: com.fasterxml.jackson:core:2.9.0
 2 Warning jackson-core has 2 vulnerabilities

 3 Downloading: ch.qos.logback:logback-classic:1.1
 4 Downloading: com.github.gumtreediff:core:2.1.0
 5 Downloading: org.scala-lang:scala-library:2.11
 6 Compiling package core: failure
+7 core/foo.java: unable to parse
 8 Total time: 5.361 s
 9 Final Memory: 19M/179M
+10 Error compiling project
```

(b) Failing log

Our dataset of build logs

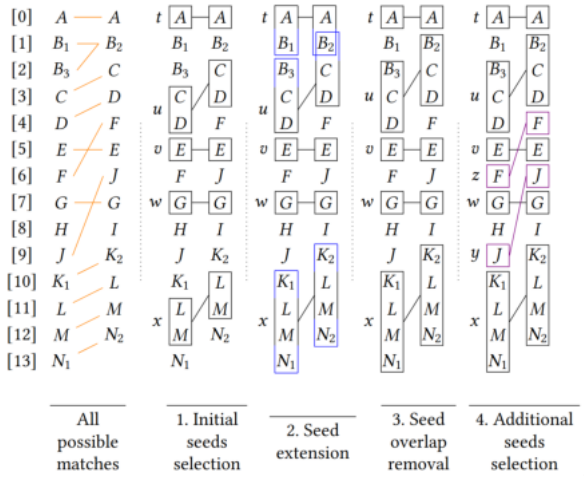


(a) Main programming language (b) Total number of lines

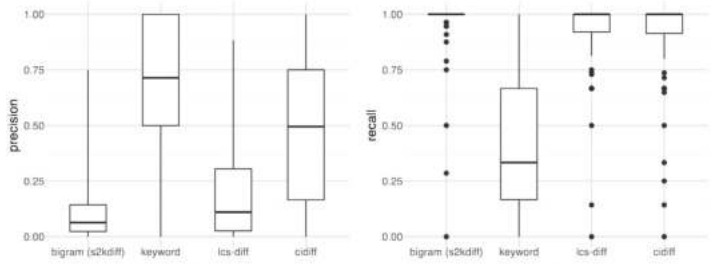
Results

- A novel heuristic to diff build logs with, in comparison to the classical diff:
  - detection of updated and moved lines
  - 30% less actions produced
  - 2 times more accurate without impact on the recall
- An open-source implementation of this algorithm with a GUI tailored to assist developpers
- A novel dataset of 17 906 pipeline regressions

The seed and extend algorithm

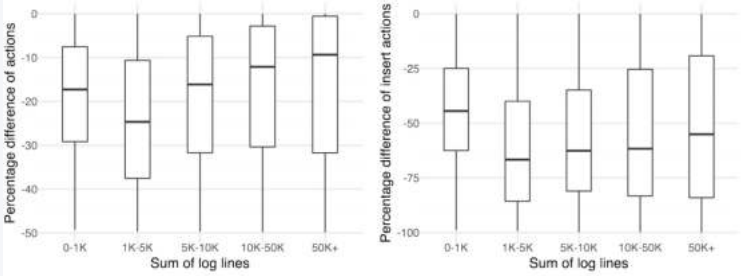


Accuracy of different strategies for pointing the error



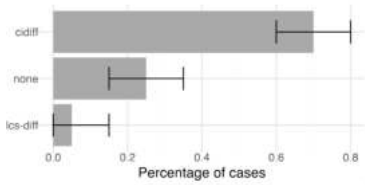
(a) Precisions (b) Recalls

Percentage difference of produced actions



(a) Size of the edit-script (b) Number of added lines

User study results



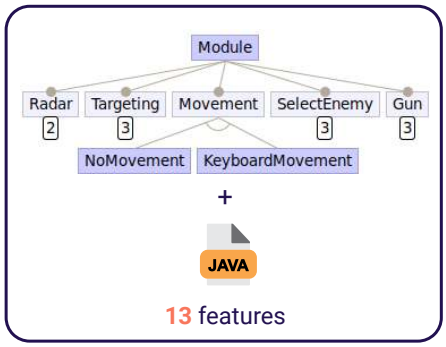
université de BORDEAUX cnrs BORDEAUX INP Inria inventeurs du monde numérique École doctorale Mathématiques et informatique université de BORDEAUX

LLM-based Co-Evolution of Configurable Software Systems

Nada Zine, Clément Quinton, Romain Rouvoy
Univ. Lille, CNRS, Inria, Centrale Lille
UMR 9189 CRIStAL, F-59000, Lille

Available soon !

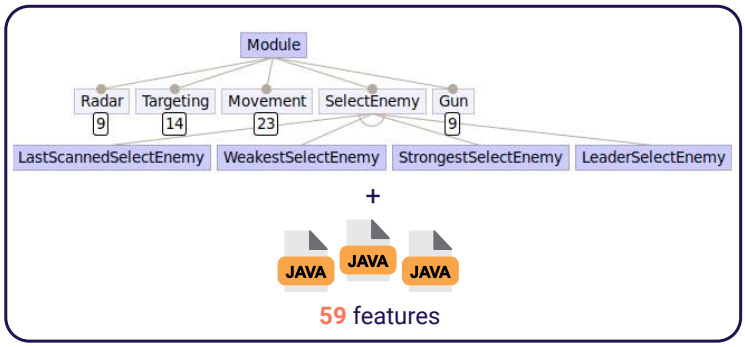
Problem: Tedious Feature Model ↔ Code Co-evolution



Project $v1$

Manual Co-evolution

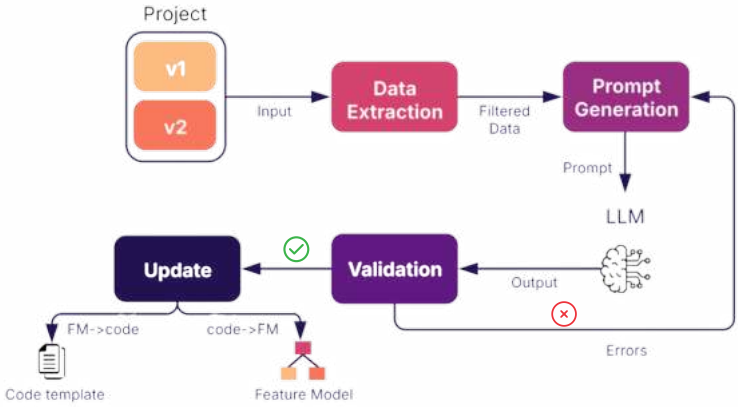
- Difficult
- Time consuming
- Error-prone



Project $v7$

Approach: LLM-based Automated Co-Evolution

Can **LLMs** support **bidirectional co-evolution** between **feature models** and **code**?



Dataset - RobocodeSPL[1]

13→59 features & 108→104.328 config

$v1$ — $v2$ — $v3$ — $v4$ — $v5$ — $v6$ — $v7$

Evolution types included[2]

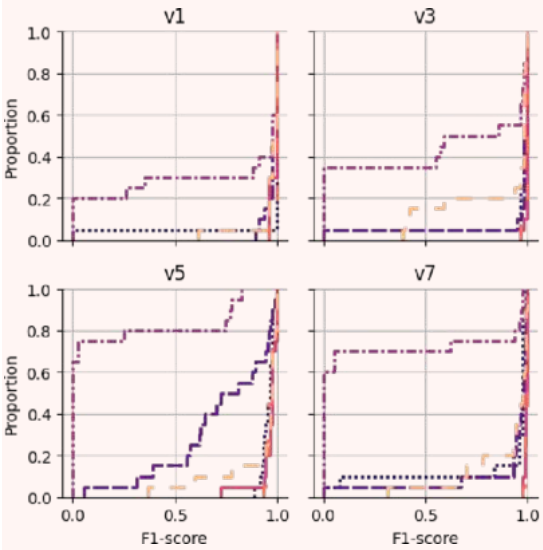
Refactoring - Generalization - Arbitrary - Specialization

Experimental Setup & Evaluation

- **6 open-source LLMs** from **3B to 32B**, leading to at least **840 prompts** per co-evolution direction.
- **F1-scores** computed using the **SFTM** tree-matching algorithm[3].
- Experiments on **Grid5000**.

- DeepSeek-R1-Distill-Qwen-14B
- Llama-3.1-8B-Instruct
- Phi-3-mini-128k-instruct
- Qwen2.5-Coder-14B-Instruct
- Qwen2.5-Coder-32B-Instruct
- Qwen2.5-Coder-7B-Instruct

Code → Feature Model



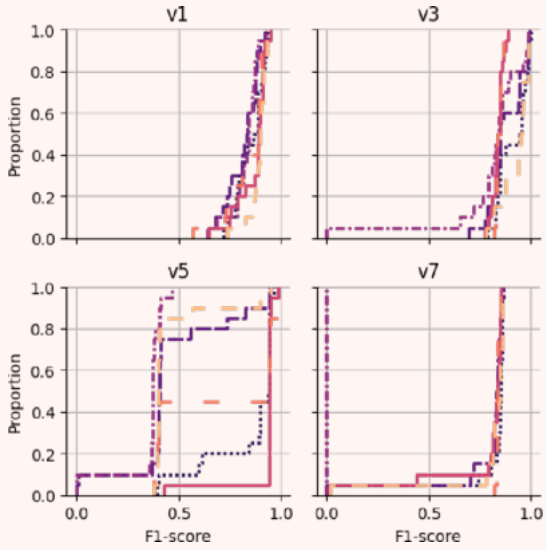
Key Findings

- **Promising results** despite some **instability.**
- **Larger models** perform **better** generally, but gains **plateau**.
- **Smaller open-source LLMs** remain **underexplored**.

Future work

- **Stabilize** LLMs (hallucination & context limits).
- More **automation** for Feature Model → Code.
- Experiments on **more complex datasets**.

Feature Model → Code



e.g. Qwen 14B 0.93→1.0

Higher performance → steeper curves shifted right

e.g. Qwen 14B 0.41→0.99

ACKNOWLEDGEMENT / REFERENCES

1.Jabier Martinez, Xhevahire Tërnava, and Tewfik Ziadi. 2018. Software product line extraction from variability-rich systems: the robocode case study. In Proceedings of the 22nd International Systems and Software Product Line Conference-Volume 1. 132–142.

2.Thomas Thum, Don Batory, and Christian Kastner. 2009. Reasoning about edits to feature models. In IEEE 31st International Conference on Software Engineering.254–264.

3.Sacha Brisset, Romain Rouvoy, Lionel Seinturier, and Renaud Pawlak. 2023. Sftm: Fast matching of web pages using similarity-based flexible tree matching.Information Systems112 (2023), 102126.

CARECloud
FRANCE 2030 PROGRAMME DE RECHERCHE CLOUD

CONTACT

Nada Zine
Univ. Lille, CNRS, Inria, Centrale Lille
UMR 9189 CRIStAL, F-59000, Lille
nada.zine@inria.fr

Chasing One-day Vulnerabilities Across Open Source Forks

Romain Lefeuvre [1] Charly Reux[1] Stefano Zacchiroli[2] Olivier Barais[1] Benoit Combemale[1]

[1] *IRISA, Université de Rennes, Inria, Diverse*

[2] *LTCI, Télécom Paris, Institut Polytechnique de Paris*

Context

Open source software plays a **critical role** in digital infrastructure and fosters **code reuse**. This reuse is enabled by two main mechanisms:

1. **Dependencies**: Importing code in a separate project.

2. **Fork**: Creating a new project from an existing one.

The security of the global open-source ecosystem attracted significant attention in recent years, particularly in the wake of high-profile software supply chain attacks on specific dependencies (e.g Log4Shell).

Although forking practices are widespread in open source ecosystem, it receives less attention. Forks share code history with their upstream repository but also potentially vulnerabilities, making it crucial to propagate vulnerability information to forks.

Current tooling failed in propagating vulnerability information to associated forks.

Challenge

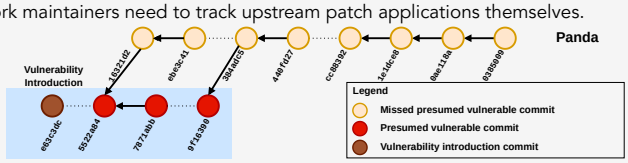
Current approaches reason locally...

1. The impacted repository is cloned.
2. Commits between the vulnerability introduction and fix are labelled as vulnerable.

Identification of vulnerable commits associated to CVE-2019-13164 in Qemu repository

... Resulting in undetected vulnerable commits in forks

1. End users or projects depending on those fork versions are not alerted.
2. Fork maintainers need to track upstream patch applications themselves.



Missed vulnerable commits in Panda-re repository, a fork of Qemu

Forks can still remain impacted by a vulnerability for which a patch exists: A one-day vulnerability

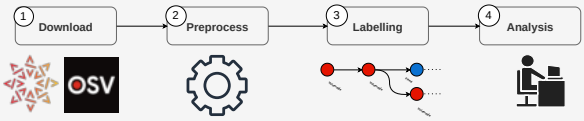
Research Methodology

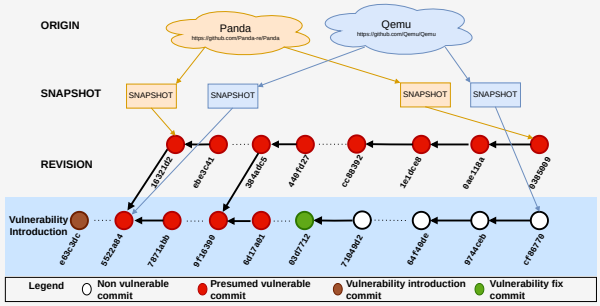
Leverage the global commit graph to track vulnerability throughout the open source ecosystem

Our approach is based on the Software Heritage (SWH) Graph, the largest public code archive, offering a global model of software source code history.

1. The SWH graph is deduplicated, a project and all its forks are linked.

2. Relying on a vulnerability database (OSV.dev) we labelled the SWH graph and propagated vulnerability information to the impacted forks.





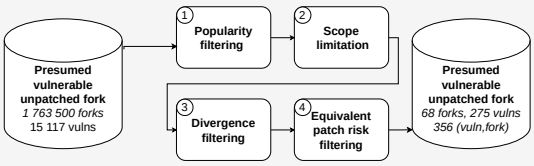
Labelling of vulnerable revision in deduplicated SWH global model

Identify one-day vulnerabilities in real-world forked open source projects

From 7162 repositories declaring vulnerabilities in OSV.dev we detected
2.2 M potential vulnerable forks detected

1. Most forks are not real world open source projects, for instance : git-flow forks or educational forks.

2. Forks might not be impacted by upstream vulnerabilities due to divergence.



Strict filtering to identify vulnerable forks

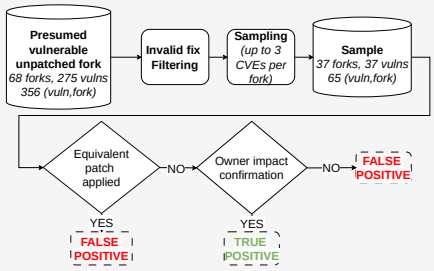
68 forks identified as potentially vulnerable **after strict filtering**

A manual evaluation has been conducted on a sample of **37 forks**

26 forks remained classified as vulnerable after manual vetting

4 one day vulnerabilities have been already confirmed as true positive by maintainer

- **panda-re/panda**, a dynamic analysis platform based on QEMU, affected by CVE-2019-13164
- **sonyxperiadev/kernel**, the Linux kernel fork by the Sony corporation, for Xperia Android devices, affected by CVE-2021-45485 and CVE-2021-4154
- **bitcoin-sv/bitcoin-sv**, a fork of bitcoin affected by CVE-2021-37492



Manual evaluation

Towards a tooling support for developers

Multi level of tooling

1. For fork project : Report on unpatched vulnerability from upstream repo

2. For project depending on a fork : Report on vulnerable fork dependencies

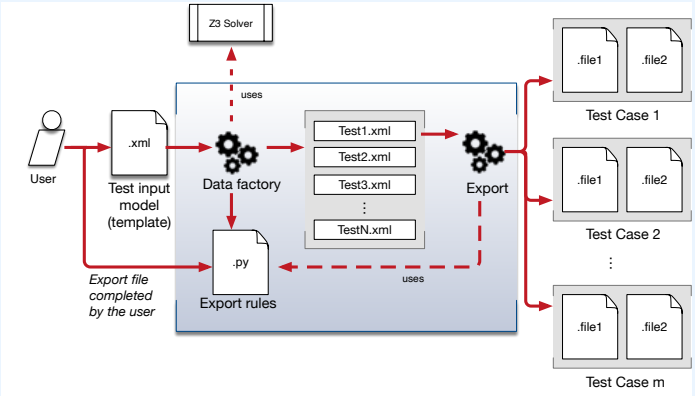
TAF : un outil de génération de cas de test supportant les structures définies récursivement

Valentin Bouziat, Jérémie Guiochet, Hélène Waeselynck
[1]Laboratoire d'Analyse et d'Architecture des Systèmes - CNRS, Toulouse
Journées Nationales du GDR GPL 2025, Pau

Introduction

TAF est un outil open source de génération de cas de test sous contrainte. À partir de la description d'un modèle de test en XML, celui-ci produit des cas de test en XML. TAF est écrit en Python et repose sur l'utilisation du solveur Z3 pour la résolution des contraintes.

Fonctionnement



L'utilisateur renseigne un fichier XML appelé **Test Model** décrivant la structure de données à générer et les contraintes définissant le système sous test.
Les cas de test sont ensuite générés en maximisant la diversité et la résolution des contraintes est effectuée par niveau de profondeur dans l'arbre XML.

Fonctionnalités notables :

- Paramétrage de l'outil dans un fichier **settings.xml**
- Définition par l'utilisateur de règles d'exportation des données (facilite l'interfaçage avec le système sous test).
- Possibilité de réutiliser des cas de test partiellement instanciés.

Le modèle de test

Eléments XML déclarables :

- **root** : un unique élément racine encapsulant tous les autres.
- **node** : pour la définition des objets.
- **parameter** : paramètres générés aléatoirement (booléen, entier, flottant ou string) encapsulés dans des nodes.
- **constraint** : contraintes sur les paramètres, utilisant possiblement la logique du premier ordre.
- **type** : nodes non instanciés, réutilisables et permettant la définition d'objets récursifs.

Exemple :

Génération de 3 objets avec :

1. un attribut entier compris entre 1 et 10
2. un attribut entier compris entre 1 et 100
3. le premier attribut doit être la racine carrée du second

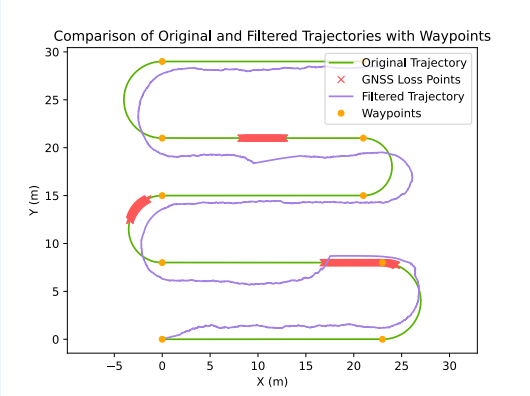
TAF Input :

```
<root name="test_case">
  <node name="my_node" nb_instances="3">
    <parameter name="la_racine"
type="integer" min="1" max="10"/>
    <parameter name="le_carre"
type="integer" min="1" max="100"/>
    <constraint name="racine_carree"
      expressions="my_node[i]\la_racine^2
EQ my_node[i]\le_carre"
      quantifiers="i"
      ranges="[0,my_node.nb_instances-1]"
      types="forall"/>
  </node>
</root>
```

TAF Output :

```
<test_case>
  <seed value="1hC41RyWw1"/>
  <mon_noeud instance="0/2">
    <la_racine>8</la_racine>
    <le_carre>64</le_carre>
  </mon_noeud>
  <mon_noeud instance="1/2">
    <la_racine>6</la_racine>
    <le_carre>36</le_carre>
  </mon_noeud>
  <mon_noeud instance="2/2">
    <la_racine>3</la_racine>
    <le_carre>9</le_carre>
  </mon_noeud>
</test_case>
```

Cas d'usage : Robotique agricole



Système sous test : Algorithmes de localisation multi-capteurs

TAF est utilisé pour la génération de :

- Waypoints,
- Biais et bruits sur les capteurs,
- Coupure de signaux.

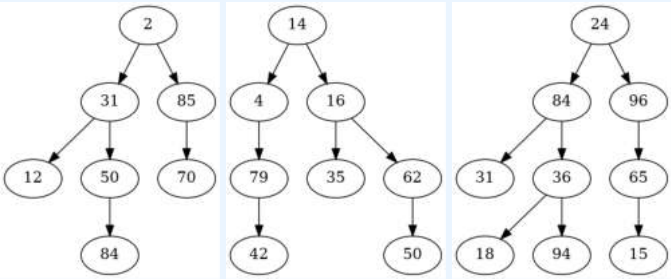
Structures définies récursivement

Test Model utilisant une structure définie récursivement pour la génération d'arbres binaires :

```
<root name="test_case">
  <type name="tree_node">
    <parameter name="key" type="integer" min="1" max="100"/>
    <node name="left_node" type="tree_node"
      min="0" max="1" min_depth="1" max_depth="3" />
    <node name="right_node" type="tree_node"
      min="0" max="1" min_depth="1" max_depth="3" />
  </type>

  <node name="tree_inst" type ="tree_node">

</root>
```

Paramétrage des structures définies récursivement :

- min et max paramètrent la largeur de la structure
- depth ou, min_depth et max_depth paramètrent la profondeur de la structure



Arbres binaires générés avec TAF

Perspectives

- Intégrer la possibilité de déclarer des contraintes entre objets avec liens de parenté dans les structures récursives (parent / enfant).
- Paralléliser la génération des cas de test, particulièrement pour les structures de données définies récursivement.

Lien utiles

Site Internet : https://wp.laas.fr/taf/

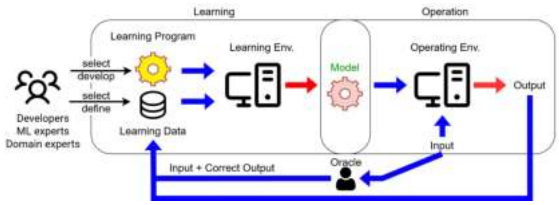
Dépôt Git : https://redmine.laas.fr/projects/taf/

Testing Online Learning Programs: A Scenario-based Approach

Maxence Demougeot, Sylvie Trouilhet, Jean-Paul Arcangeli, Françoise Adreit
{FirstName.LastName}@irit.fr
Institut de Recherche en Informatique de Toulouse, Université de Toulouse, UT2J

CONTEXT AND PROBLEM

Online Machine Learning



Online Machine Learning and Operation

Why?
- Unanticipated data
- Need of updating the model continuously

How?
- Based on repeated comparisons between produced and correct outputs (what the outputs should be) [1]
- When the model produces an output, an oracle gives the correct output as feedback to the learning program
  => Learning and operation phases are mixed

Testing ML-based Systems

Like traditional software, their behavior must be verified. Testing allows to:
- Identify favorable/unfavorable use cases
- Highlight the presence of a defect
- Compare versions
- Evaluate parameter settings

The literature focuses on offline ML and model testing [2,3]

We focus on testing online learning programs

CONTRIBUTIONS [4]

Testing Online Learning Programs

Does the program learn "well" throughout its scope of use?
⇔ Does it build "good" models based on training data?

$\forall D \in Scope, P1(D) \Rightarrow P2(M_D)$

D: training data set
Scope: set of all possible training data sets
P1: property of the training data set
P2: expected property of the produced model M_D with data D

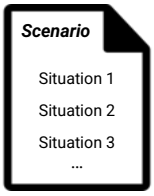
Example
IF D contains no routes that use the corridor 3 **(P1)**,
THEN to go from point A to point B, M_D guides the robot without passing through corridor 3 **(P2)**

Issues
- Non-deterministic nature
- No reference model for comparison (oracle problem)
- When is the model ready for testing?
- Expensive activity

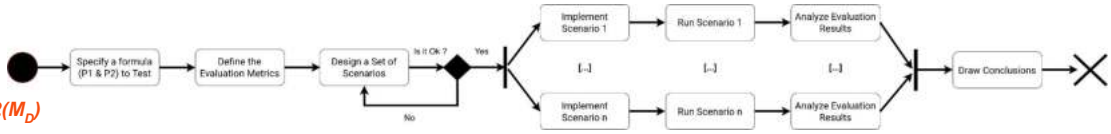
Research Questions

- **RQ1**: How to **design** test cases that include **learning and evaluation steps** and conform to the **expected properties**?
- **RQ2**: How to automate **generation** and **selection** of a test data set that **sufficiently covers** the expected scope of the learning program?
- **RQ3**: How to organize the **testing workflow**, from the **test data set design** to **execution** and **evaluation**?
- **RQ4**: As part of the workflow, how to **implement** a test case, **automate** its execution, and **determine** if the test passes or not?

Scenario-based Testing Approach



$\forall D \in Scope, P1(D) \Rightarrow P2(M_D)$



Activity diagram of the scenario-based testing process

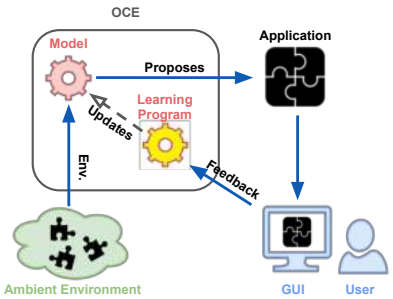
Principle: Indirect Evaluation of the Learning Program
1. Make it build a certain model from D that conforms to P1
2. Put the model in situations to evaluate if its behavior satisfies P2

A scenario is a story describing a sequence of:
- **Learning situations**
  Situation = input + correct output
- **Evaluation situations**
  Situation = input + expected property of the produced output

Design by:
- Domain experts
- Based on their business knowledge

RQ1&3

Scenario-based Testing of OCE

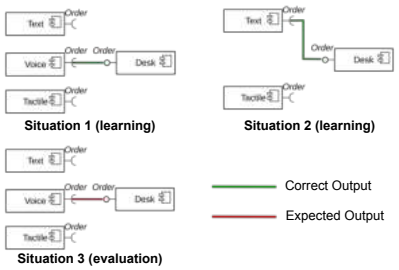


Opportunistic Composition Engine (OCE)
- Dynamically configures the user's ambient environment according to their preferences and the software components that are present in the environment
- Learns online the user's preferences
- OCE's learning program gets feedback from the user to update the model

Goal: Check that OCE learns well the user's preferences
⇒ Check that OCE proposes the "right" application at the "right" time

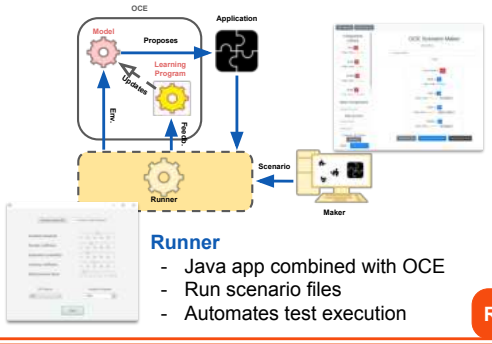
Toy Scenario Example

IF the user expresses a preference for a component that disappears **(P1)**
THEN if it reappears, OCE proposes an application including that component again **(P2)**



Tooling

Maker
- VueJS app
- Scenario design by hand
- Produces JSON scenario files



Runner
- Java app combined with OCE
- Run scenario files
- Automates test execution

RQ4

Automatic Generation of Scenarios

Designing test scenarios is a complex task
- Expensive
- Prone to oversights

Hard to guarantee a good coverage of the application domain

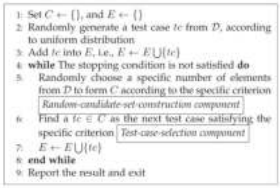
The human expert defines a region of the input space by expressing:
- A scenario model (a formula to be tested)
- Domain data (e.g., software components)

⇒ Generation of scenarios in the region

Automatic Selection of Scenarios

Potentially to many scenarios to test
⇒ How to select a set of representative scenarios?

Adaptive Random Testing of Scenarios
Select Test From Candidates Strategy (STFCS) based on a distance metric between scenarios
1. Randomly generate a set of candidate scenarios
2. Select the scenario furthest away from those already selected


1: Set $C \leftarrow \{\}$, and $E \leftarrow \{\}$
2: Randomly generate a test case tc from $\mathcal{D}$, according to uniform distribution
3: Add tc into E, i.e., $E \leftarrow E \bigcup \{tc\}$
4: **while** The stopping condition is not satisfied **do**
5: Randomly choose a specific number of elements from $\mathcal{D}$ to form C according to the specific criterion *Random-candidate-set-construction component*
6: Find a $tc \in C$ as the next test case satisfying the specific criterion *Test-case-selection component*
7: $E \leftarrow E \bigcup \{tc\}$
8: **end while**
9: Report the result and exit

Select Test From Candidates Strategy Framework [5]

RQ2

References:
[1] RUSSELL, Stuart J. et NORVIG, Peter. *Artificial intelligence: a modern approach*. Pearson, 2016.
[2] ZHANG, Jie M., HARMAN, Mark, MA, Lei, *et al.* Machine learning testing: Survey, landscapes and horizons. *IEEE Transactions on Software Engineering*, 2020, vol. 48, no 1, p. 1-36.
[3] RICCIO, Vincenzo, JAHANGIROVA, Gunel, STOCCO, Andrea, *et al.* Testing machine learning based systems: a systematic mapping. *Empirical Software Engineering*, 2020, vol. 25, p. 5193-5254.
[4] DEMOUGEOT, Maxence, TROUILHET, Sylvie, ARCANGELI, Jean-Paul, *et al.* Scenario-based Testing of Online Learning Programs. *20th International Conference on Software Technologies (ICSOFT 2025)*, 2025.
[5] HUANG, Rubing, SUN, Weifeng, XU, Yinyin, *et al.* A survey on adaptive random testing. *IEEE Transactions on Software Engineering*, 2019, vol. 47, no 10, p. 2052-2083.

Deductive Verification of Synchronous Reactive Programs

Térence Clastres, Frédéric Dabrowski

Université d'Orléans, INSA CVL, LIFO, UR 4022, Orléans, France

Deductive Verification

- Formal method to verify the functional correctness of programs
- Starts from an assumption (*precondition*) $\{P\}$ about the program initial state and tries to conclude the last state reached is within an accepted set of states $\{Q\}$ (postcondition)
- Uses deductively applied inference rules to produce mathematical statements requiring a proof
- Specification is written in a logic language (e.g. predicate logic)
- Various tools developed to ease the verification process, from SMT solvers (Alt-Ergo, Z3…), to dedicated verification platforms communicating with them (Why3, Dafny…)

Inference Rules

$$\frac{}{\{P[e/x]\}\ x := e\ \{P\}}\ assign$$

$$\frac{\{P\}\ c_1\ \{Q\} \qquad \{Q\}\ c_2\ \{R\}}{\{P\}\ c_1; c_2\ \{R\}}\ seq$$

Annotated Program

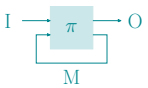
$\{a = v_a \wedge b = v_b\}$

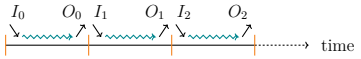
```
c := a;
a := b;
b := c
```

$\{a = v_b \wedge b = v_a\}$

Synchronous Reactive Programs

- Continuous interaction with the environment
- Infinite execution
- Logic clock, notion of *instant*





For our purposes, programs are written in a simple imperative language with arduino-like setup/loop procedures:

```
input on off : bool;
output light : bool;

setup : x := false;
loop:
  if on then
    emit true to light;
  else if off then
    emit false to light; end
  end
```

Temporal Properties

- Instead of viewing a program as a transformational machine, focus on its behaviour over time
- Decomposition Theorem: every program property is a conjunction of a safety and liveness property.

Safety: a bad thing never happens

- e.g. loop invariant
- proof tool: induction
- doing a bad thing is irremediable
  → finite counter-example

Liveness: a good thing eventually happens

- e.g. program termination
- proof tools: well-founded relations, coinduction
- a good thing can always be done later
  → infinite counter-example

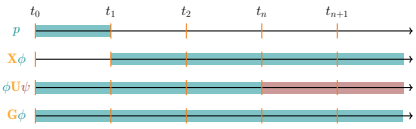
Linear Temporal Logic

- Need for a concise way to describe a program behaviour
- A behaviour is seen as an infinite word σ (e.g. abababababa…)

Syntax

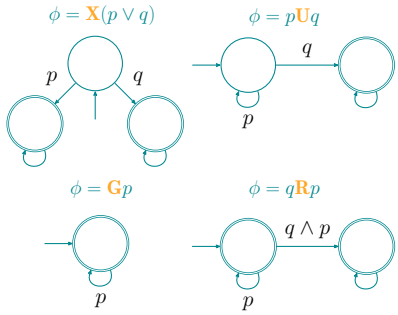
$$\mathbf{LTL}(\Sigma) \ni \phi ::= p \in \Sigma \mid \neg\phi \mid \phi \vee \phi \mid \mathbf{X}\phi \mid \phi\mathbf{U}\phi \mid \mathbf{G}\phi \mid \phi\mathbf{R}\phi$$

Semantics



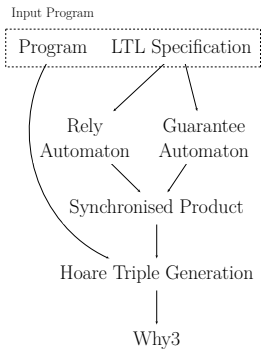
Büchi Automaton

- Nondeterministic finite-state automaton on infinite words (recognizer)
- A run is an infinite sequence of state and transitions, starting from a state marked 'initial'
- A word is read left to right. To each letter x must correspond a transition labelled x from the current state to the next
- No final state but an acceptance condition: certain states are marked 'acceptant', and at least one of them must be present infinitely many times in the run for it to be acceptant
- Effective construction from an LTL formula
- More expressive than LTL (equivalent to ω-regular expressions)



Verification Procedure for Safety Properties

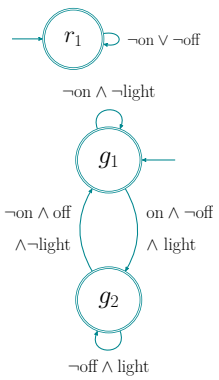
Overview



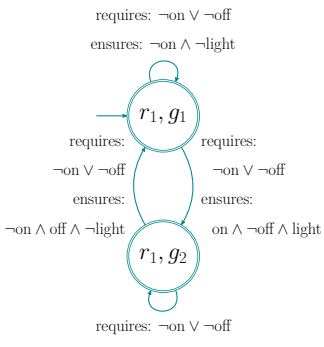
Input Program

```
input on off : bool;
output light : bool;

relies_on G (!on || !off).

guarantees
  on R (on || !light).
guarantees
  G (on -> off R (off || light)).
guarantees
  G (off -> on R (on || !light)).

setup :
  ensures { x = false }
  x := false;

loop:
  if on then
    emit true to light;
  else if off then
    emit false to light; end
  end
```

Specification Automata



Synchronized Product



Output Why3 Program

```
module Program
  (* ... *)
  let setup = begin
    ensures { outputs.light = false }
    outputs.light <- false  end

  let r1g2 = begin
    requires {(not (hd _history)._i.off /\ (outputs.light)) \/
        ((hd _history)._i.on /\ not (hd _history)._i.off /\
          ↪  outputs.light)}
    requires { not inputs.on \/ not inputs.off }
    ensures { (not inputs.off /\ outputs.light)
                \/
                (not inputs.on /\ inputs.off /\ not
                  ↪  (outputs.light)) }
    if inputs.on then outputs.light <- true
    else (if inputs.off then outputs.light <- false) end

  let r1g1 = begin
    requires {(not (hd _history)._i.on /\ not outputs.light) \/
                (length _history = 0 /\ outputs.light = false)\/
      (not (hd _history)._i.on /\ (hd _history)._i.off /\ not
        ↪  outputs.light)}
    requires { not inputs.on \/ not inputs.off }
    ensures {(inputs.on /\ not inputs.off /\ (outputs.light))
                \/ (not inputs.on /\ not outputs.light) }
    if inputs.on then outputs.light <- true
    else (if inputs.off then outputs.light <- false) end
end
```

Future Work

Nondeterministic Specification

- Adds more flexibility
- Büchi automata are ∃-automata: a word is accepted if at least one path is acceptant
- Our method requires all paths to be acceptant
- This approach is correct but too naive: useless constraints are generated
  → refinement of the specification automaton

Liveness

- Safety properties enable us to mark every state acceptant
- Liveness requires showing there are infinitely many accepting states along a path
- How to deal with liveness in rely formula? → fairness assumptions

Internal Memory

- Specification needs to mention the program internal memory for successful verification
- Internal memory is an implementation detail that shouldn't be visible
- Write a public and private specification and show the private specification implies the public one

Multiple Programs

- Currently, a single program
- Goal: multiple programs in cooperation
- Parallel / sequential composition
- Controlled memory sharing
- Need for compositionality

IoT-Based Smart Mobility Framework for Stress Detection in ASD Drivers

Amadou Djoulde Barry (LAAS-CNRS), Nawal Guermouche (LAAS-CNRS)
Viviane Kostrubiec (CERPPS), Pierre Vincent paubel (CLLE)

context and Objective

- **ASD** people faces challenges
  - Anxiety and panic attacks
  - Sensosy-motory disorders
  - Lack of communication and social skills
- **Modern cars and roads** are primarily designed for people with typical neurodevelopment
- **Individuals with ASD** experience higher Stress and reduced driving performance compared to neurotypical drivers (Cox et al.,Fok et al)
- **Objective:** Develop an IoT-Based Smart mobility framework to predict and mitigate stress triggers in ASD drivers before they occur.

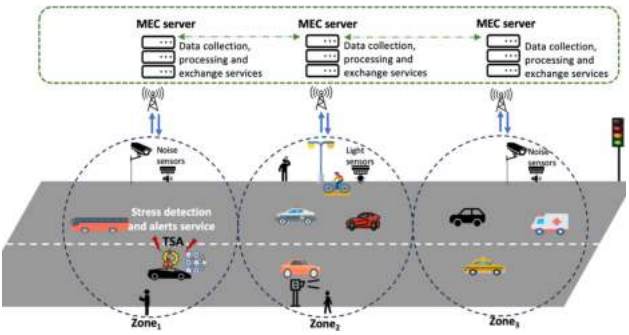
1. Cox et al. pilot study comparing newly licensed drivers with and without autism and experienced drivers in simulated and on-road driving. Journal f Autism and Developmental Disorders 50, 1258 – 1268 (2020)
2. Fok et al. perceived driving difficulty, negative affect, and emotion dysregulation in self-identified autistic emerging drivers. Frontiers in Psychology 13 (2022)

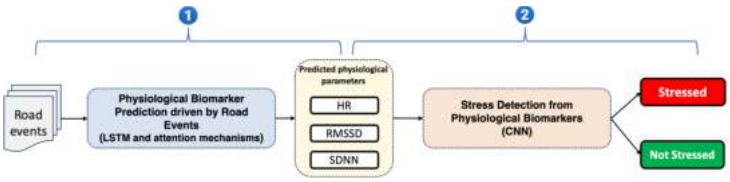
Related Work

- Current solutions(sensory-friendy environments, wearable technologies, virtual reality simulators) only **react** or provide training they don't predict stress beforehand (Benssassi et al., Bian et al.)
- **Existing stress detection systems** for drivers are reactive, identifying stress only after it manifests. (Amin et al.)
- Our study aims to provide **proactive** approach **predicting stress** induced by roads events in advance

1. Benssassi et al.Wearable assistive technologies for autism: Opportunities and challenges. IEEE Pulse 9(6), 39–42 (2018)
2. Bian et al. Design of a physiology-based adaptive virtual reality driving platform for individuals with asd. ACM Transactions on Accessible Computing (TACCESS) 12, 1 – 24 (2019)
3. Amin et al. ecg based driver's stress detection using deep transfer learning and fuzzy logic approaches. IEEE Access 10, 29788–29809 (2022)

Proposed Approach





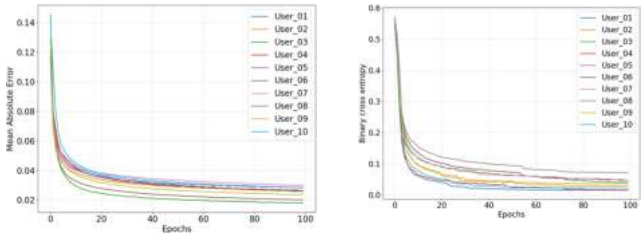
- **IoT Sensor:** installed along roadsides and in vehicles, collect real-time environmental data
- **Multi-Access Edge Computing:** Streams **road-event** and **Context data** (traffic density, noise light, weather) directly to the vehicle .
- **In car AI model**: Analyse synchronized physiological and contextual. data to anticipate upcoming stress and issue real-time alerts when evated stresss is detected

- We propose a two-stage based stress detection:
  - **LSTM and attention based model** for biomarkers prediction (HR, RMSSD, SDNN)
  - **CNN-based stress classification of predicted biomarkers**

Dataset acquisition

- **Driving platform**: SimulAuto 360° setup with Renault Zoe cabin equipped with SCANeR™studio for developing driving scenario
- **ECG acquisition:** ECG signal was collected via two Ag/AgCl electrodes using the actiCHamp system at 500HZ
- **Study participants:** Ten participants (7 women, 3 men; aged 22-23)

Results



- **Biomarker forecasting with LSTM-attention**: The model achieved average MAE = 0.025 and MSE = 0.0023 on the training data.
- **Stress classification:** CNN attained mean accuracy = 98.7%, recall = 98.3%, AUC = 99.8%.

Discussions and perspectives

Predicting stress before it occurs can improve road safety

- **Study limitations:** Small sample size (N = 10) and limited road diversity in simulations.
- **Future work**: Expand data collection, enhance model performance, and explore new stress quantification methods.

LAAS-CNRS
/ Laboratoire d'analyse et d'architecture des systèmes du CNRS

Laboratoire conventionné avec
Université de Toulouse

On the use of generalisation and unification for composing interactions via gate connection

Christian Joel Kenne Nguetoum[1,2]
christian.nguetoumkenne@cea.fr

Advisors: Pascale Le Gall [1], **Boutheina Bannour**[2]

1 Université Paris-Saclay, CentraleSupélec, Laboratoire de Mathématiques et d'Informatique pour la Complexité et les Systèmes (MICS), 91192 Gif-Sur-Yvette, France

2 CEA Paris Saclay, Centre Nano-Innov, Laboratoire d'Exigences et de Conformité des Systèmes(LECS), 91120 Palaiseau, France

Interaction models with gates

We use the Interaction Language [1] to specify distributed systems. In addition to its algebraic structure, it provides a easy-to-read, diagrammatic notation similar to MSCs and UML Sequence Diagrams.
Each subsystem is represented by a vertical line called a **lifeline**, with message exchanges shown as horizontal arrows between lifelines. The language includes scheduling operators to express complex behaviours: **vp** for message passing, **loop** for repetition, **par** for concurrency, **alt** for choice, and **seq** for weak sequencing.

Actions correspond to message emissions (outgoing arrows) and receptions (incoming arrows), specifying communication with the environment (**env**). When an emission in one model and a reception in another refer to the same message, they are called **complementary actions**. Such actions are linked via a shared object known as a **gate**. In the example below, the emission of the message auth by the lifeline client (client!auth[1] in model *i*) and its reception by broker (broker?auth[1] in model *j*) are complementary and share the gate 1.

$i = \mathsf{seq}\big(\mathsf{client!auth}[1],\ \mathsf{alt}(\mathsf{client?failure}[3],\ \mathsf{client?success}[2])\big)$ $\quad j = \mathsf{seq}\big(\mathsf{broker?auth}[1], \mathsf{seq}(\ldots)\big)$

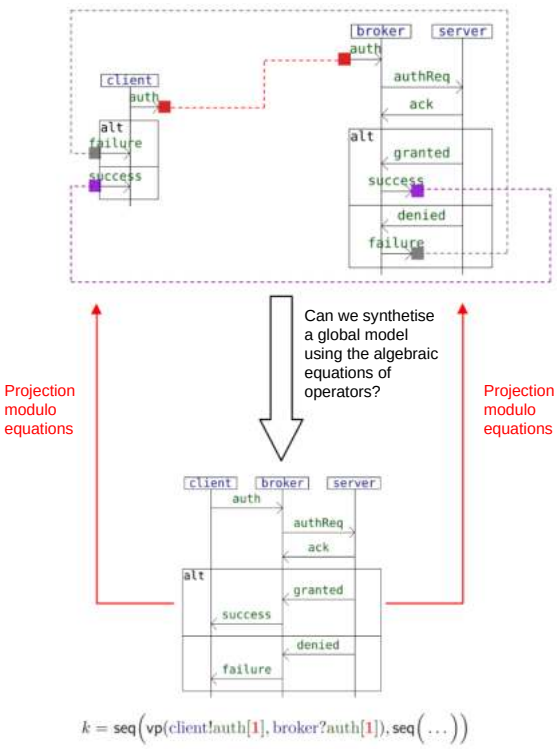
The operators obey algebraic laws grounded in the formal semantics [1] of the language.

Property	Operator	Equation
Associativity	$\mathsf{f} \in \{\mathsf{seq}, \mathsf{par}, \mathsf{alt}\}$	$\mathsf{f}(\mathsf{f}(x,y),z) \approx \mathsf{f}(x,\mathsf{f}(y,z))$
Commutativity	$\mathsf{f} \in \{\mathsf{par}, \mathsf{alt}\}$	$\mathsf{f}(x,y) \approx \mathsf{f}(y,x)$
Idempotence	alt	$\mathsf{alt}(x,x) \approx x$
Unit element $\varnothing$	$\mathsf{f} \in \{\mathsf{seq}, \mathsf{par}\}$	$\mathsf{f}(x,\varnothing) \approx x, \mathsf{f}(\varnothing,x) \approx x$
$\varnothing$-fixpoint	loop	$\mathsf{loop}(\varnothing) \approx \varnothing$

Table. Basic set *E* of algebraic equations of the Interaction Language

Interaction composition

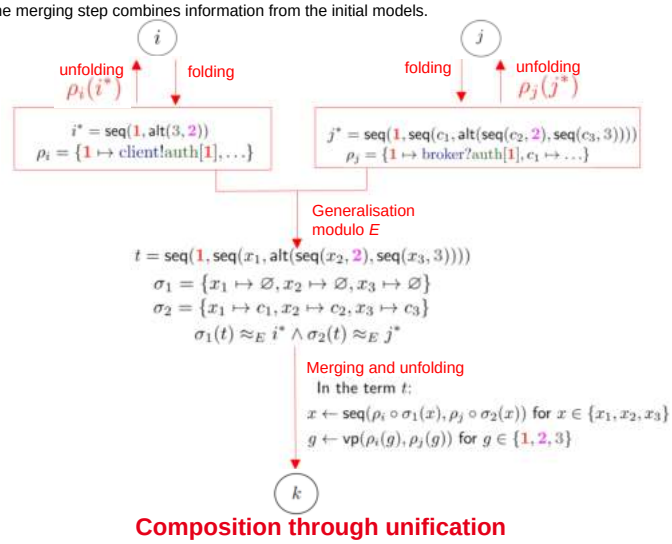
Model composition involves synthesising a new model by gluing complementary actions, those linked by a shared gate into message passings (**vp**), while preserving the scheduling information of the original models. A valid composition must **project** [2] back to the initial models from which it was constructed.
In our example, the emission and reception of auth are composed into the message passing **vp**(client!auth[1],broker?auth[1]) in the model *k*.



$k = \mathsf{seq}\big(\mathsf{vp}(\mathsf{client!auth}[1], \mathsf{broker?auth}[1]), \mathsf{seq}(\ldots)\big)$

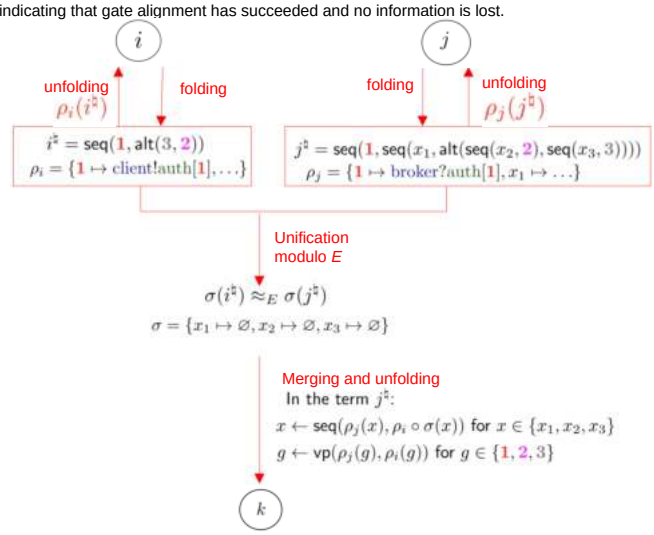
Composition through generalisation

The first step is **model folding**, which hides subterms that contain no actions relevant to the composition inside fresh constants, and replaces actions with their associated gates. The resulting folded terms are significantly simpler and are then generalised modulo *E* [3]. **The desired generaliser must include all relevant gates**, indicating successful alignment of complementary actions. Such a generaliser is said to be **well-formed**. Finally, the merging step combines information from the initial models.



Composition through unification

The first step is model folding, where actions are replaced by their associated gates. Unlike the previous approach, irrelevant subterms are hidden using fresh variables. The second step is unification modulo *E* [4]. **The desired unifier maps variables to subterms that contain no gates, and ensures that applying it to one term yields a structure containing all the variables of the other**. These two conditions define what we call a **well-formed** unifier, indicating that gate alignment has succeeded and no information is lost.



Conclusion

We have explored two methods for composing interactions using algebraic equations. These approaches are not interchangeable, as there are cases where a well-formed generaliser exists while no well-formed unifier can be found. We are therefore interested in delineating the scope of each method and their respective ability to decide when composition is possible. Future work will focus on formalising correctness guarantees and investigating algorithmic aspects.

[1] Mahe, Erwan. An Operational Semantics of Interactions for Verifying Partially Observed Executions of Distributed Systems. Phdthesis, Université Paris-Saclay, 2021.

[2] Julien Lange and Emilio Tuosto. Synthesising choreographies from local session types. In 23rd international conference on Concurrency Theory (CONCUR'12), 2012.

[3] M. Alpuente, S. Escobar, J. Meseguer, and J. Sapina. Order-sorted equational generalization algorithm revisited. Annals of Mathematics and Artificial Intelligence, 2022.

[4] S. Escobar, J. Meseguer, and R. Sasse. Variant Narrowing and Equational Unification. Electronic Notes in Theoretical Computer Science, 2009.

MODÈLES EXÉCUTABLES POUR LA SIMULATION

Hugo RICHARD, Jean-Philippe BABAU, Éric CARIOU

hugo.richard@univ-brest.fr, jean-philippe.babau@univ-brest.fr, eric.cariou@univ-brest.fr

CONTEXTE & PROBLÉMATIQUE

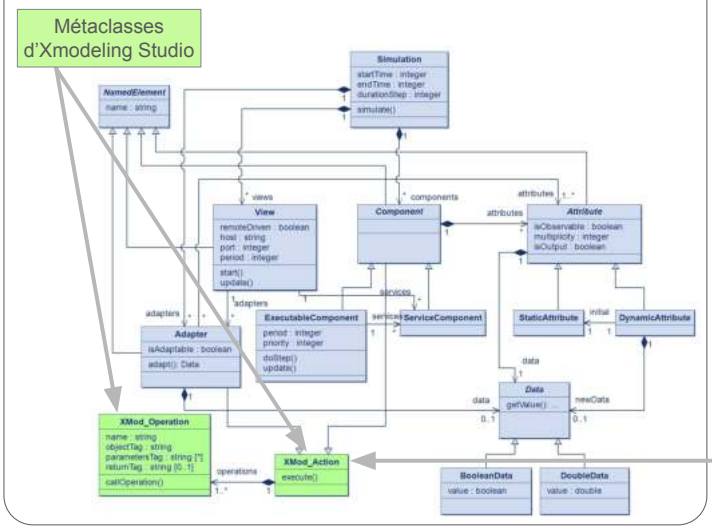
- Dans un logiciel : un comportement qui pilote l'appel d'opérations métier
  - Ex : quand j'appuie sur le bouton Marche, on suit une transition qui fait chauffer le four
- En IDM, un DSL dont les modèles sont exécutables (xDSL) permet de spécifier ce comportement à un haut niveau d'abstraction
  - Moteur d'exécution : implémente la sémantique d'exécution du xDSL
  - Le modèle et le moteur d'exécution peuvent être embarqués dans le logiciel final

- Outils existants :
  - **Xmodeling Studio [1]** : permet de rajouter automatiquement des opérations métier Java à des éléments d'un xDSL et d'appeler ces opérations pendant l'exécution
    - Limite : manque de typage et de liaison entre le modèle et le code
  - **CARES [2]** : génère le code d'une simulation à partir de DSL d'architecture de composants et d'un scénario de simulation
    - Limite : le lien avec les opérations métier de simulation est à écrire à la main

Objectifs de la thèse :
- **Proposer une meilleure liaison générique entre les modèles exécutables et les opérations métier**
- **Appliquer cette solution à CARES pour faciliter le développement de simulateurs logiciels**

INGÉNIEUR DE LANGAGE : DÉFINIT LE xDSL ET SON MOTEUR D'EXÉCUTION

MÉTAMODÈLE DE CARES



MOTEUR D'EXÉCUTION

- Exécution pas-à-pas des composants selon leur fréquence et priorité avec diffusion instantanée
- Le moteur fait le lien avec les opérations gérées par XModeling Studio

```
// class Simulation
public void simulate() {
    List<ExecutableComponent> ecs = new ArrayList<>();
    for (Component c : components) {
        if (c instanceof ExecutableComponent) ecs.add((ExecutableComponent) c);
    }
    ecs.sort(Comparator.comparingLong(ExecutableComponent::getPriority));

    for (View v : views) v.start();

    for (long i = startTime; i < endTime; i++) {
        for (ExecutableComponent ec : ecs) {
            if (i % ec.getPeriod() == 0) {
                ec.doStep();
                ec.update();
            }
        }
    }
}
// class ExecutableComponent
public void doStep() {
    // exécute automatiquement les opérations métier associées aux composants
    // grâce aux fonctionnalités de XModelingStudio
    this.execute();
}
```

INGÉNIEUR LOGICIEL : DÉFINIT LE MODÈLE ET LES OPÉRATIONS MÉTIER

MODÈLE

- Création d'un modèle de drone avec une **position**, un niveau de **batterie** et une **vue**
- Paramétrage de l'appel du code métier

```
Simulation drone(0,100,1000) {
    Executable pos(1,1) { ... };

    Executable bat(2,1) {
        observable double capacity(1075) : out dynamic level;

        call calcNextBatLevel("this.attributes[level].value", "const double 40",
            "const double 0.987", "const double 0.35", "const double 1")
            on "ext.calcs" result "this.attributes[level].newvalue";
    };

    Adapter batPercent(true) {
        in bat.level, bat.capacity;
        out double;
        call toPercent("bat.attributes[level].value","bat.attributes[capacity].value")
            on "ext.calcs" result "this.data.value";
    };

    remote View viewUnity("localhost",2223,1000) {
        log "batPercent.data.value";
        ...
    };
}
```

Nom de la méthode

Paramètres

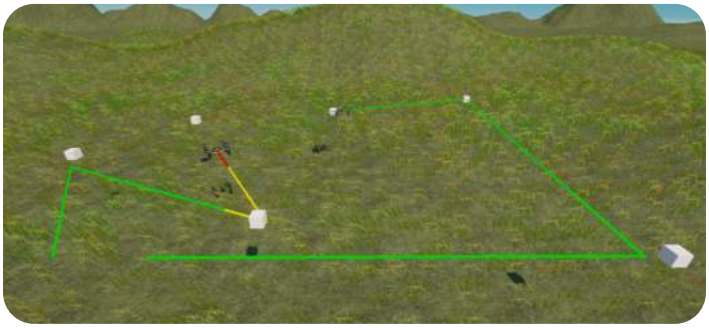
Appel à une fonction métier

Retour

Objet d'appel de la méthode

CODE MÉTIER : FONCTIONS DE SIMULATION

```
public class DroneSimulationFunctions {
    /**
     * Calculates the next battery level after a given duration.
     * @param level of battery in MJ
     * @param speed in m/s
     * @param mass in kg
     * @param efficiency the engine efficiency (between 0 and 1)
     * @param duration in second
     * @return the next battery level in MJ
     */
    public double calcNextBatLevel(double level, double speed, double mass, double efficiency, long duration) {
        double power = ExtUtils.calculatePower(efficiency,mass,speed);
        double consumedEnergy = ExtUtils.energyConsumption(power,(duration/1000.0)) / 1_000_000; // converted in MJ
        return Math.max(level - consumedEnergy, 0);
    }

    /**
     * Converts a given level to a percentage.
     * @param level of battery in MJ
     * @param capacity of battery in MJ
     * @return the percentage of battery between 0 and 100
     */
    public double toPercent(double level, double capacity) {
        return level / capacity * 100;
    }
}
```



LANCEMENT DE LA SIMULATION

- Instanciation des objets métier extérieurs
- Chargement du modèle de simulation
- Association du modèle et des objets métier
- Appel de la fonction simulate() pour lancer la simulation

```
public static void main(String[] args) {
    DroneSimulationFunctions calcs = new DroneSimulationFunctions();
    HashMap<String, Object> map = new HashMap<>();
    map.put("calcs", calcs);
    Simulation sim = SDUtils.loadSimulation(MODEL);
    SDUtils.setMap(map);

    sim.simulate();
}
```

BILAN & CHALLENGES

- CARES facilite aujourd'hui la programmation de la **sémantique opérationnelle** et permet l'**interprétation** du modèle
- Xmodeling Studio permet un lien **agnostique** du méta-modèle et du langage vers des opérations métier

1. Renforcer Xmodeling Studio par le **typage** et des **contrats** sur les opérations
2. Étendre le xDSL de CARES pour la **structuration** et l'**instanciation**
3. Appliquer ces principes au niveau méta : pouvoir changer les méthodes du moteur pour changer la **sémantique d'exécution**

[1] Eric Cariou, Olivier Le Goaër, Léa Brunschwig, Franck Barbier: A generic solution for weaving business code into executable models. EXE Workshop at MoDELS 2018, CEUR Workshop Proceedings, vol. 2245, October 2018
[2] Loïc Salmon, Pierre Yves Pillain, Goulven Guillou, Jean-Philippe Babau: NAVIDRO, a CARES architectural style for configuring drone co-simulation. ACM Trans. Embed. Comput. Syst. 23(3): 39:1-39:26 (2024)

SkeltyMLOps: A Reference Architecture for Collaborative MLOps

EuroMov
Digital Health in Motion

IMT Mines Alès
École Mines-Télécom

Authors

Charbel DAOUD[1]
Charbel.Daoud@mines-ales.fr

Danielle AZAR[2]
Danielle.Azar@lau.edu.lb

Julie BOICHÉ[1]
Julie.Boiche@umontpellier.fr

Christelle URTADO[1]
Christelle.Urtado@mines-ales.fr

Sylvain VAUTTIER[1]
Sylvain.Vauttier@mines-ales.fr

1: EuroMov Digital Health in Motion, Univ. Montpellier & IMT Mines Alès, Alès, France
2: Department of Computer Science & Mathematics, Lebanese American University, Byblos, Lebanon

Acknowledgements

The PLANETS PhD is partly funded by Région Occitanie.

1. Introduction and Context

- **DevOps** accelerates software delivery via automated build, test, and deployment pipelines [7].
- Machine Learning (ML) adoption is growing rapidly, leading to a strong emphasis on its operationalization.
- To bridge the gap between ML development and operation, DevOps principles are being adapted to ML **giving rise to MLOps (Machine Learning Operations)**.

MLOps still faces some **challenges**:

- Lack of standardized reference architectures [4],
- Complex integration of diverse technologies and tools [3],
- High cost and black box commercial solutions [2],
- Inconsistent definitions across the literature [5].

A reference architecture designed to promote collaboration between the diverse actors involved in MLOps processes is needed.

2. Research Questions

- **RQ1**: Who are the main collaborating actors involved in MLOps?
- **RQ2**: What are the dimensions of MLOps?
- **RQ3**: What are the different activities that constitute MLOps processes?
- **RQ4**: How should an MLOps architecture be designed to facilitate collaboration among the actors identified by RQ1 and effectively integrate the activities identified by RQ3?

3. LLM-Powered SLR Methodology

To answer RQ1, RQ2, and RQ3, a **Systematic Literature Review (SLR)** was performed. The SLR was powered by **a Large Language Model (LLM)** to assist in **clustering** and generating names for the **actors** and **activities** and to reduce **human bias.** The OpenAI o3-model was used in this study. The SLR process is illustrated below:

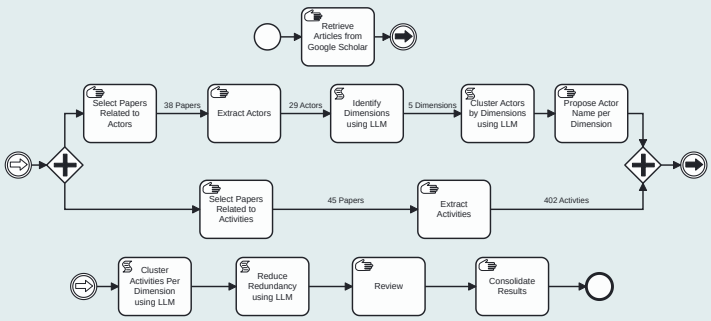


Fig. 1 - Overview of the SLR Methodology

4. MLOps Actors, Dimensions, and Activities

The results of RQ1, RQ2, and RQ3, derived from the SLR, are illustrated in the figure below.

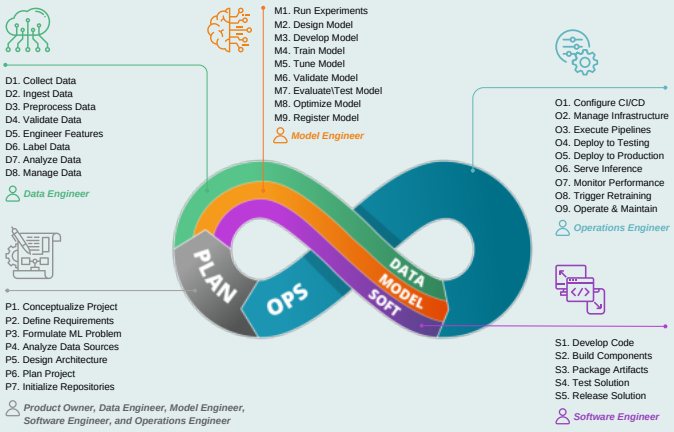


Fig. 2 - The MLOps infinite loop illustrating the dimensions, actors, and associated activities

5. A Reference MLOps Architecture Promoting Collaboration

Based on the identified Activities, Actors, and Dimensions, **SkeltyMLOps**, a collaborative reference architecture for MLOps (addressing RQ4), is designed and illustrated below.

Key characteristics of SkeltyMLOps:

- **Project Management**: coordinating business goals, development tasks and issuing tickets
- **MLOps Process Orchestrator**: executing and adjusting processes

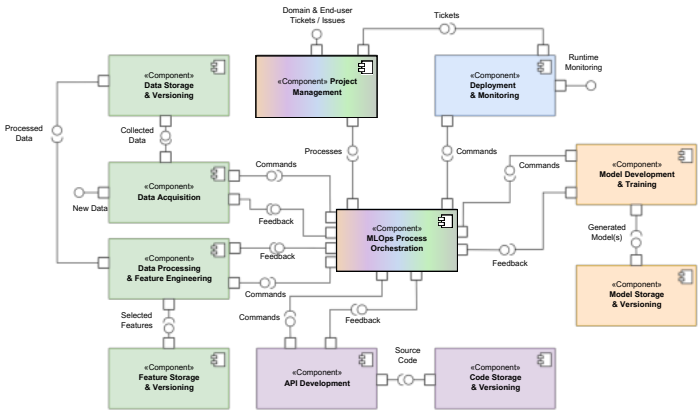


Fig. 3 - Component Diagram of SkeltyMLOps

6. State of the Art in MLOps Architectures

Paper	Identified Gaps	How SkeltyMLOps Addresses Them
Wozniak et al. [6]	Lacks clearly defined data management components.	Introduces structured and explicit data management components.
Najafabadi et al. [1]	Lists 35 architectural components without clearly defining responsibilities or interactions. Notes the need for an orchestrator but omits details.	Clearly defines actors and their interactions, and integrates a central orchestrator into the architecture.
Kreuzberger et al. [5]	Omits comprehensive data versioning for raw and training/test data. Monitoring doesn't catch data drift or infrastructure issues.	Provides complete data versioning across all stages and integrates structured monitoring with feedback loops.

7. On-going Work and Perspectives

- Design generic collaborative **MLOps processes** and their variants using BPMN,
- Design a BPMN-based **orchestrator** for SkeltyMLOps,
- Implement a **full prototype** of SkeltyMLOps built with widely adopted tools and libraries,
- **Evaluate** SkeltyMLOps across various continuous-training case studies.

References

[1] F. Amou Najafabadi, J. Bogner, I. Gerostathopoulos, P. Lago, An analysis of MLOps architectures: A systematic mapping study, in: Proc. of 18th European Conference on Software Architecture (ECSA), volume 14889, Springer, Luxembourg City, Luxembourg, 2024, pp. 69–85.
[2] A. M. Burgueño-Romero, A. Benítez-Hidalgo, C. Barba-González, J. F. Aldana-Montes, Towards an open-source MLOps architecture, IEEE Software 42 (2025) 59–64
[3] F. Calefato, F. Lanubile, L. Quaranta, A preliminary investigation of MLOps practices in GitHub, in: 16th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement, 2022, pp. 283–288.
[4] A. Kolar Narayanappa, C. Amrit, An analysis of the barriers preventing the implementation of MLOps, in: IFIP WG 8.6 International working conference on Transfer and Diffusion of IT (TDIT), Springer, 2023, pp. 101–114.
[5] D. Kreuzberger, N. Kühl, S. Hirschl, Machine learning operations (mlops): Overview, definition, and architecture, IEEE access 11 (2023) 31866–31879.
[6] A. P. Woźniak, M. Milczarek, J. Woźniak, MLOps components, tools, process and metrics - a systematic literature review, IEEE Access 13 (2025) 22166–22175.
[7] Zhu, L. Bass, G. Champlin-Scharff, DevOps and its practices, IEEE software 33 (2016) 32–34.

Scan for Prompts & Queries.

Toward Adaptive MLOps: Variability Mapping and Modeling

EuroMov
Digital Health in Motion

IMT Mines Alès
École Mines-Télécom

Authors

Chadine El Hatimi [1]
chadine.el-hatimi@etu.mines-ales.fr

Vincent Berry[2]
vincent.berry@umontpellier.fr

Abdelhak Djamel Seriai [2]
abdelhak.seriai@lirmm.fr

Valère Preney [3]
valere@getcaas.io

Xavier Quesnot[3]
xavier@getcaas.io

Chouki Tibermacine[4]
chouki.tibermacine@univ-ubs.fr

Christelle Urtado[1]
Christelle.Urtado@mines-ales.fr

Sylvain Vauttier[1]
Sylvain.Vauttier@mines-ales.fr

1: EuroMov Digital Health in Motion, Univ. Montpellier & IMT Mines Alès, Alès, France
2: LIRMM, Univ. Montpellier, CNRS, Montpellier, France
3: GetCaaS, https://www.getcaas.io/
4: Université Bretagne Sud, UMR CNRS 6074, IRISA, Vannes, France

References

[1]Symeonidis, Georgios, et al. "MIOps definitions, tools and challenges." 2022 IEEE 12th Annual Computing and Communication Workshop and Conference (CCWC). IEEE, 2022.
[2]Osterweil, L. (2011). Software Processes are Software Too. In: Tarr, P., Wolf, A. (eds) Engineering of Software, pp 323--344. Springer. https://doi.org/10.1007/978-3-642-19823-6_17
[3]Terenciani, Marcelo, Débora Maria Barroso Paiva, Geraldo B. Landre, and Maria Istela Cagnin. "BPMN*-A Notation for Representation of Variability in Business Process Towards Supporting Business Process Line Modeling." In SEKE, pp. 227-230. 2015.
[4]Colantoni, Alessandro, Luca Berardinelli, and Manuel Wimmer. "DevopsML: Towards modeling devops processes and platforms." Proceedings of the 23rd ACM/IEEE International Conference on Model Driven Engineering Languages and Systems: Companion Proceedings. 1st LowCode Workshop.pp 1-10. 2020

Acknowledgements

AdaptiveMLOps (ANR-24-IAS2-0004) is an ANR-funded research project. The contribution presented here was conducted as part of a **Master's thesis** in AI and Data Science

anr

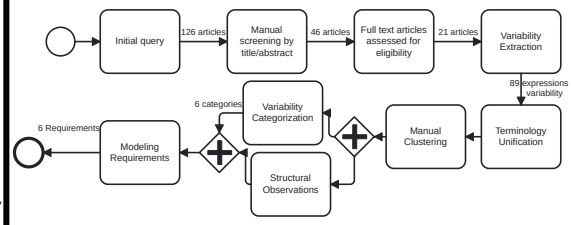
1-Designing Configurable MLOps Pipelines

The increasing adoption of **MLOps**-*"the collection of techniques and tools for deploying machine learning models in production"* [1]-has led to a wide diversity of pipelines, varying in structure, tools, and execution strategies. This diversity introduces significant **variability.** To address this, we build on the concept of **Software Process Line (SPrL)**, a paradigm that extends **Software Product Line Engineering (SPLE)** to process modeling under the rationale that *"software processes are software too"* [2]. SPrLs provide a systematic approach to manage variability across families of processes, making them a promising foundation for modeling adaptive MLOps pipelines. Based on these foundations, our research is guided by two questions:

- **RQ1:** What types of variability are present in existing MLOps pipelines?
- **RQ2:** How can model-driven engineering be leveraged to model MLOps variability?

2-From MLOps Variability Mining to Modeling Needs

We conducted a lightweight **systematic mapping study** starting from *126* papers. After screening and eligibility assessment, *21* papers were selected and analyzed, resulting in *89* expressions of variability.



Overview of the research process

Through terminology unification-grouping semantically similar expressions (e.g., "data drift handling" and "dataset shift detection") under unified terms- and manual clustering, we derived two key outcomes: a categorization of variability types and structural observations on how variability appears in process and architecture views. These outcomes were integrated to derive our modeling requirements.

A-Variability Categories

Behavioral	Variability in the order, and execution flow of pipeline activities. This includes triggers, retraining logic, and conditional execution.
Functional	Variability in what the pipeline is designed to do — e.g., task type, algorithm types
Technical / Tooling	Variability in the choice of platforms, frameworks, libraries, infrastructure...
Non-Functional	Variability in aspects like transparency, performance, explainability, sustainability...
Organizational	Variability in compliance requirements, domain regulations, or ethical policies
Domain-Specific	Variability that depends on the specific domain of application (e.g., IoT vs. healthcare)

B-Structural Observations : Process and Architecture Views

After analyzing the variability types we extracted, we also noticed a layered pattern in MLOps wokflows: MLOps systems are constructed from approaches (e.g. CI/CD, manual execution, scheduled execution), which determine the pipeline type. Each pipeline is composed of steps and utilizes tools and assets (e.g., data, models, platforms).

Pipelines are often connected together via triggers, events, and metrics, forming inter-pipeline workflows. These workflows can be analyzed and modeled through two interconnected perspectives:

- **Process View:** Represents activity sequences, triggers...
- **Architecture View:** Represents tools, infrastructure and components to execute the pipeline.

For this reason the design of MLOps pipelines must account for ***both the behavior of workflows and the underlying structure***. A complete modeling solution should capture and integrate both the process logic and the architecture.

C-Modeling Requirements

We identified requirements that reflect both **what** needs to vary (based on our six variability categories) and **where** that variability must be supported (process vs architecture view). The requirements define the capabilities that our modeling solution should support:

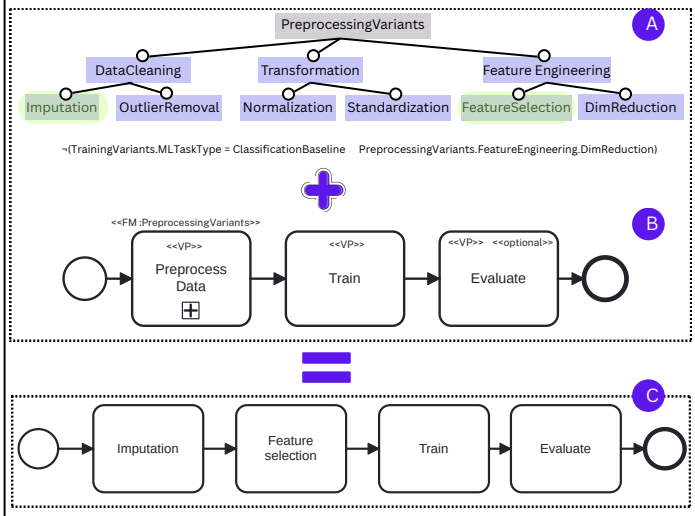
- **R1** - Support for process control flow and conditional execution
- **R2** - Support for optional and alternative pipeline steps
- **R3** - Support for component-level composition and modularity
- **R4** - Support for reactive orchestration
- **R5** - Support for multi-dimensional variability
- **R6** - Support for domain-specific extensions
- **R7** - Support for constraints and dependencies
- **R8** - Support for configuration resolution

3-Functional Variability Illustrative Example: (ongoing work)

R1,R2,R3,R7

The diagram illustrates the process view of a configurable MLOps pipeline, where functional variability is modeled using a **BPMN***-inspired notation [3]. This approach follows the principles of a **Software Process Line** (SPrL), where a family of process variants is managed through a shared base model with configurable elements.

- Each variation point (e.g., Preprocess, Evaluate) corresponds to a configurable subprocess whose internal structure may differ depending on the selected features.
- These variations are defined in separate **feature models (A)** that we link to the process **(B)** variation points (VP) with a FM annotation.
- At configuration time, a set of feature selections determines which subprocesses are included **(C)**.



4-Perspectives

While the previous example focuses on functional variability in the process view, other categories of variability remain to be explored and formally represented. **The architecture view** too would be relevant for capturing technical and tooling variability, such as alternative platforms and services used in MLOps pipelines. A possible direction is to explore existing frameworks like **DevOpsML** [4], which already weaves together process and architecture models. However, it does not currently support variability modeling.

Models for Distributed and Dynamically Controlled Cyber Physical Systems

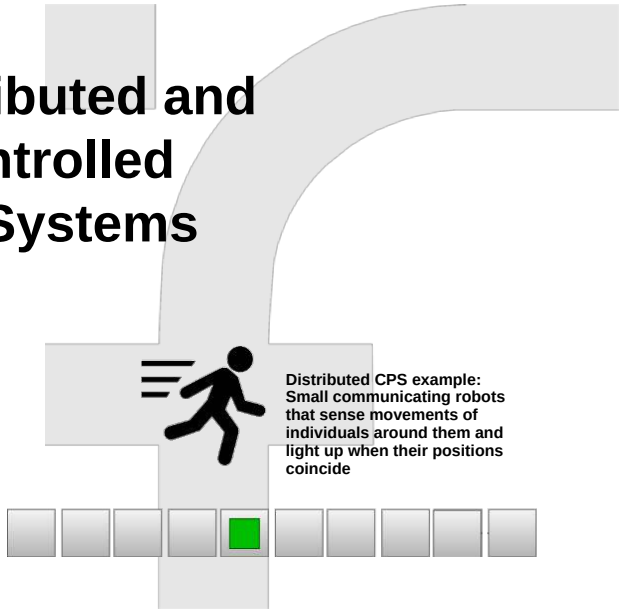
Context

Distributed Cyber Physical Systems (CPSs): sets of communicating devices pairing computation and physical processes.

CPSs must ensure many adaptation properties while running: network unavailability resistance, aberrant values tolerance, timing constraints…

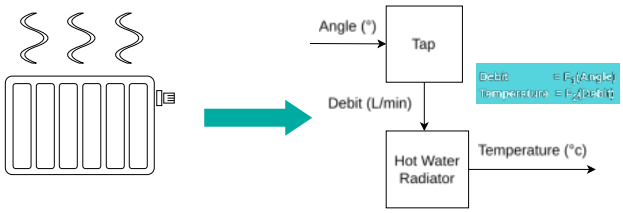
They also come with inherent challenges: complex coordination mechanisms, low memory, weak computing capacities, energy consumption…

Model formalisms help handling these aspects by automatizing the process of turning complex designs into robust implementations.

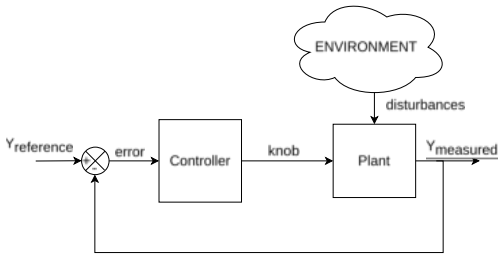


Control Theory

When it comes to adaptation, an efficient way to dynamically modify a system's behavior is to use **Control Theory** (CT) [0]. Systematic transformation of physical components into functional block model.

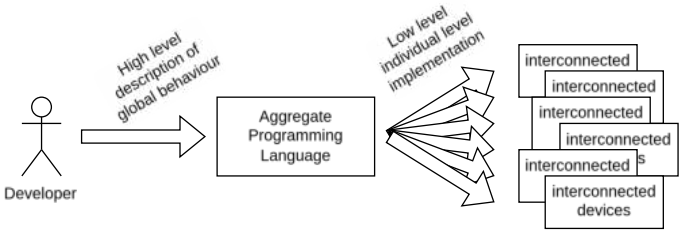


To achieve adaptation, design of feedback loops allows run-time control of the system's parameters.



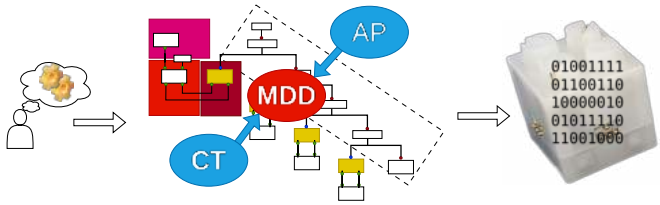
Aggregate Programming

Handling multiple CPSs requires to introduce coordination into our control theory models. Principles from **Aggregate Programming** (AP) [1] can be used to factorize complex protocols in our system descriptions.



Goal

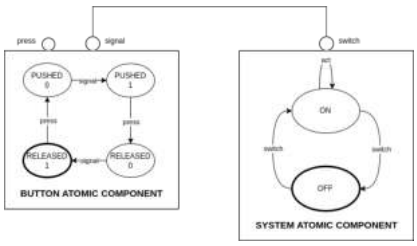
Design a full Model Driven Development (MDD) pipeline for the generation of resilient distributed CPS applications.



Approach

Bottom-up empirical construction of the pipeline tools:

- Design of example applications to implement;
- Description of a model to simulate the system behavior within the BIP framework [2] with interconnected Finite State Machines (FSM);



- Analysis of recurrent model parts and re-factorization of these components within a new model description language including AP and CT constructs, the Chips language;

{chips}

- Implementation of ATL rules [3] to automatically turn Chips models into BIP models, thus allowing the simulation, model-checking and implementation of described application.

References:
[0] Joseph L Hellerstein, Yixin Diao, Sujay Parekh & Dawn M Tilbury (2004): Feedback control of computing systems. John Wiley & Sons
[1] Mirko Viroli et al. From distributed coordination to field calculus and aggregate computing(2019): Journal of Logical and Algebraic Methods in Programming. doi: 10.1016/j.jlamp.2019.100486
[2] Ananda Basu, Saddek Bensalem, Marius Bozga et al. (2011): Rigorous Component-Based System Design Using the BIP Framework. IEEE Softw. doi:10.1109/MS.2011.27
[3] Frédéric Jouault, Freddy Allilaire, Jean Bézivin & Ivan Kurtev (2008): ATL: A model transformation tool.Science of Computer Programming, doi:10.1016/j.scico.2007.08.002

Contacts
Anna Gallone — anna.galdel@protonmail.com
Simon Bliudze — simon.bliudze@inria.fr
Sophie Cerf — sophie.cerf@inria.fr
Olga Kouchnarenko — olga.kouchnarenko@femto-st.fr

Besançon, Belfort, Montbéliard, Sévenans
www.femto-st.fr

An Interactive Tool for Goal Model Construction using a Knowledge Graph

AUTHORS:
Shahin ABDOUL SOUKOUR, PhD Student, Inria Paris
William ABOUCAYA, Research engineer, Université PSL
Nikolaos GEORGANTAS, Researcher, Inria Paris

AFFILIATIONS: Inria, PSL Université Paris

OUR PAPER

CONTEXT

- The **goal model** is fundamental in **Goal-Oriented Requirements Engineering**
  - Describes system goals hierarchically, refining high-level goals into specific ones

CHALLENGE

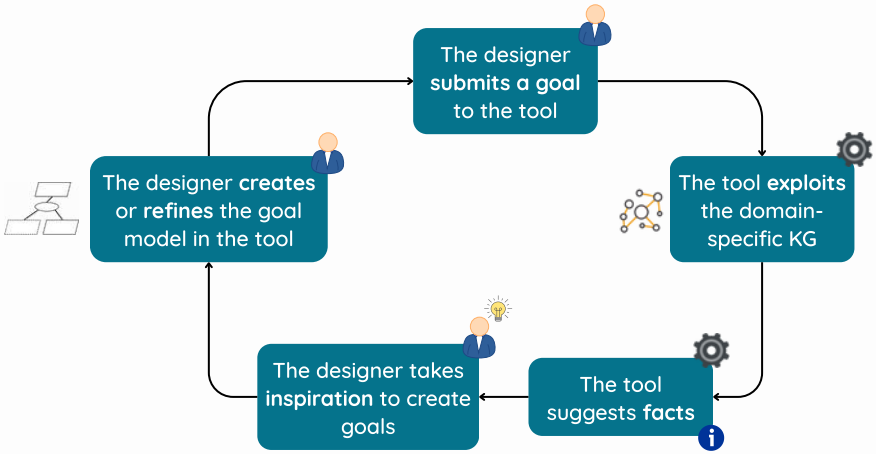
Manual goal modeling for new applications is **time-consuming**, and **complex**

SOLUTION

Interactive graphical tool for leveraging existing domain knowledge in software system design

RESEARCH TOPICS

- **Interactive design**: Allows online step-by-step refinement of goals
- **KG knowledge extraction**:
  - Semantic similarity
  - Natural Language Inference (NLI)
  - Sentiment analysis
  - Graph-to-Text (G2T) generation
- **Restricted KG exploration**: Limits KG exploration depth and width for efficiency



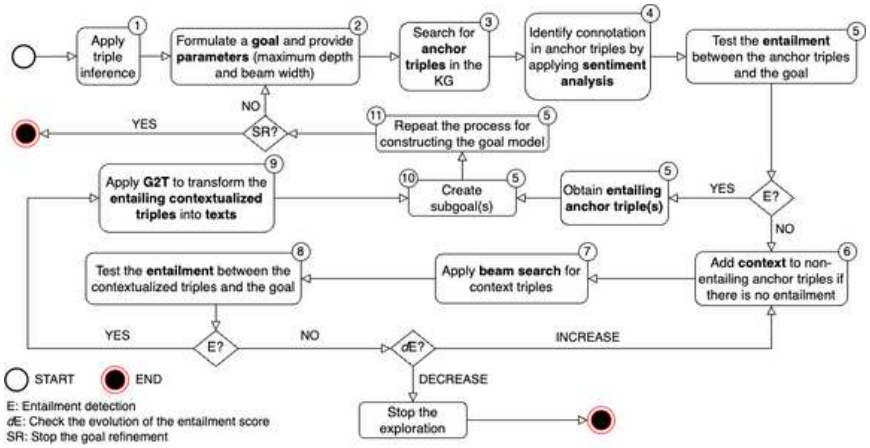
CONCLUSION

- Facilitates goal model creation interactively
- Balances exploration efficiency and relevance

NEXT STEPS

- Empirical evaluation of the tool
  - To assess the degree of facilitation for the designer and the quality of the resulting goal models

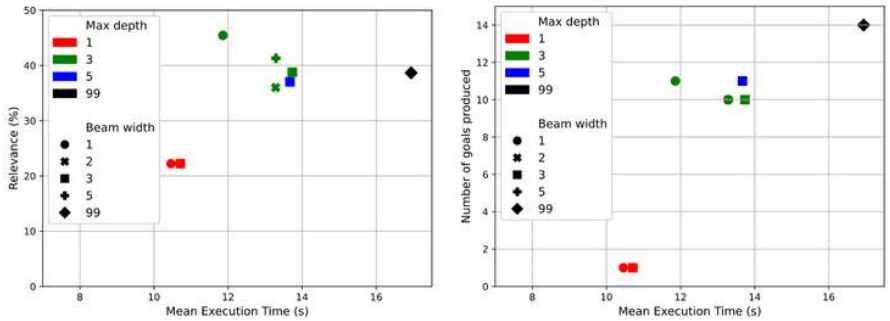
TOOL WORKFLOW FOR GOAL MODELING



EVALUATION

- Scenario: **Flood management system**
- KG details: 90 triples covering flood causes, detection techniques, impacts, and mitigation measures
- Generated **8 goal models** by varying 2 core parameters: **max depth** and **beam width** of KG exploration
- Hardware: RTX 3080, 32GB
- Results:
  - Optimal configuration: max depth = **3**, beam width = **1**
  - Relevance: 45.45%
  - Execution time: ~11.86 seconds

Evaluate the trade-off between time efficiency and quality of the resulting goal models when bounding KG exploration



Impact of the maximum depth and the beam width on:

(a) Relevance vs. time performance

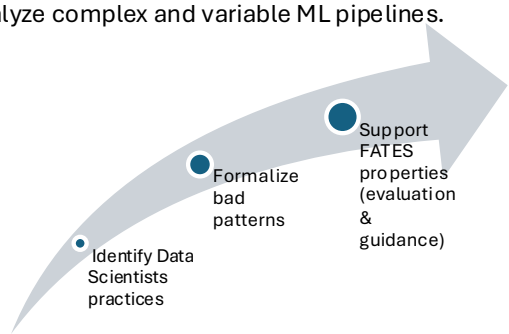
(b) Produced goals vs. time performance

ML pipelines profiling for variability-aware exploration and FATES evaluation

Nicolas LACROIX (I3S)

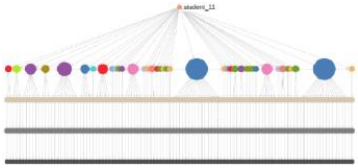
I. Context & Objectives

- Fairness, Accountability, Transparency, Ethics and Safety (FATES) compliance in ML requires providing guidelines and evaluating data scientist practices.
- Data Science is exploratory leading to diverse and evolutive workflows.
- We lack systematic and reliable processes to analyze complex and variable ML pipelines.

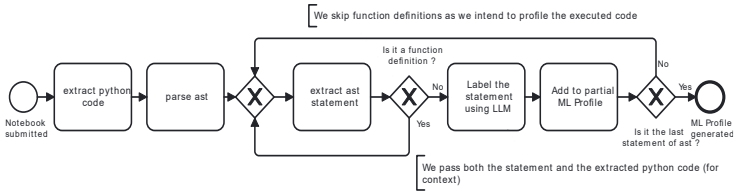


II. Reverse-engineering ML pipelines

- *MLProfile*: a layered ML pipelines abstraction to capture both organizational high-level components (e.g., data preparation) and finer-grained variations in method choices (e.g., algorithm selection).

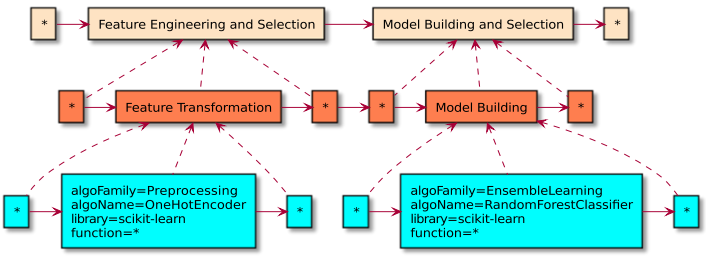


- It is constructed automatically from Python code using a hybrid LLM-rule-based parser.



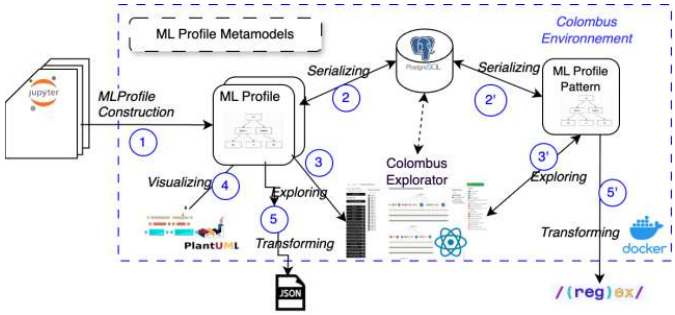
III. Pattern-driven characterization

- *MLProfile Pattern Matching (PPM)*: an operable representation characterizing ML practices.
- PPM leverages the different layers of the MLProfile.
- It benefits from composition to effectively organize, and reuse acquired knowledge.



IV. Exploration platform

- *Colombus*: a platform for exploring MLProfiles and extracting knowledge from a set of ML pipelines.
- It supports querying through pattern matching and saving experiments, including MLProfiles and patterns.



V. First Results

XP: Exploring the variability of solutions to a common problem

- Using Colombus, we identified common and outlier practices in a set of 22 notebooks from students addressing the same problem.

RESULTS OF POSITIVE AND NEGATIVE MATCHING OF THE 12 PATTERNS (*Qi*) ON THE 22 PROFILES (*Si*)

	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q8_R	Q9	Q10	Q11
S0	yes	yes	no	yes	yes	yes	no	no	yes	yes	no	yes
S1	yes	yes	no	yes	yes	no	no	no	yes	yes	no	yes
S2	yes	yes	no	yes	no	yes	yes	no	yes	no	yes	yes
S3	yes	yes	no	yes	yes	no	no	no	yes	no	no	no
S4	yes	yes	no	yes	yes	no	no	no	yes	no	yes	yes
S5	yes	no	no	yes	yes	yes	no	no	no	no	yes	no
S6	yes	yes	no	yes	yes	yes	no	no	yes	no	yes	no
S7	yes	no	yes	yes	yes	yes	yes	no	no	no	no	no
S8	yes	yes	no	yes	yes	yes	no	no	yes	no	yes	no
S9	yes	yes	no	yes	no	yes	yes	no	yes	no	yes	no
S10	yes	yes	no	yes	no	yes	yes	no	yes	no	yes	no
S11	yes	no	no	yes	yes	yes	no	no	no	no	yes	no
S12	yes	yes	no	yes	no	yes	yes	yes	yes	no	no	yes
S13	yes	yes	no	yes	yes	yes	no	no	yes	no	yes	no
S14	yes	yes	no	yes	no	yes	yes	no	yes	no	no	no
S15	yes	yes	no	yes	no	yes	yes	no	yes	no	yes	yes
S16	yes	yes	no	yes	no	yes	yes	no	yes	no	yes	no
S17	yes	yes	no	yes	yes	yes	no	no	yes	no	yes	yes
S18	yes	no	no	yes	yes	yes	no	no	no	no	yes	no
S19	yes	yes	no	yes	no	yes	yes	no	yes	no	no	no
S20	yes	yes	no	yes	yes	yes	no	no	yes	no	no	no
S21	yes	yes	no	yes	yes	no	no	no	yes	yes	yes	yes

XP: Identifying a first remediation pattern

- Benefiting from data scientist feedbacks and the exploration platform, we identified a first mitigation pattern consisting of a sequence of Model Training steps associated to the use of Fairness-related metrics.

Contacts and Acknowledgment

Nicolas Lacroix: nicolas.lacroix@univ-cotedazur.fr
Mireille Blay-Fornarino: mireille.blay@univ-cotedazur.fr
Frédéric Precioso: frederic.precisoso@univ-cotedazur.fr

ANR FATES-MLOps: ANR-24-IAS2-0002-01 (TSIA 2024)
ANR PROFIL: ANR-24-CE25-1286-01

UNIVERSITÉ CÔTE D'AZUR · i3S sophia antipolis · Inria · CNRS GDR Groupement de recherche GPL Génie de la programmation et du logiciel · anr agence nationale de la recherche