RESEARCH CENTRE

Sophia Antipolis - Méditerranée

IN PARTNERSHIP WITH:

CNRS, Université Côte d'Azur

2020 ACTIVITY REPORT

Project-Team KAIROS

Multiform Logical Time for Formal Cyber-Physical System Design

IN COLLABORATION WITH: Laboratoire informatique, signaux systèmes de Sophia Antipolis (I3S)

DOMAIN

Algorithmics, Programming, Software and Architecture

THEME

Embedded and Real-time Systems

Contents

Project-Team KAIROS

Creation of the Project-Team: 2017 January 01

Keywords

Computer sciences and digital sciences

A1.1.1. – Multicore, Manycore

A1.1.2. – Hardware accelerators (GPGPU, FPGA, etc.)

A1.2.5. – Internet of things

A1.2.7. – Cyber-physical systems

A1.5.2. – Communicating systems

A2.2. – Compilation

A2.3. – Embedded and cyber-physical systems

A2.4. – Formal method for verification, reliability, certification

A2.5.1. – Software Architecture & Design

Other research topics and application domains

B5.1. – Factory of the future

B5.4. – Microelectronics

B6.1. – Software industry

B6.4. – Internet of things

B6.6. – Embedded systems

B6.7. – Computer Industry (harware, equipments...)

B7.2. – Smart travel

B8.1. – Smart building/home

B8.2. – Connected city

B9.5.1. – Computer science

1 Team members, visitors, external collaborators

Research Scientists

- Robert de Simone [Team leader, Inria, Senior Researcher, HDR]
- Luigi Liquori [Inria, Senior Researcher, HDR]
- Eric Madelaine [Inria, Researcher, HDR]
- Dumitru Potop Butucaru [Inria, Researcher, HDR]

Faculty Members

- Julien Deantoni [Université Côte d'Azur, Associate Professor, HDR]
- Nicolas Ferry [Université Côte d'Azur, Associate Professor, from Sep 2020]
- Frederic Mallet [Université Côte d'Azur, Professor, HDR]
- Marie-Agnès Peraldi Frati [Université Côte d'Azur, Associate Professor]
- Sid Touati [Université Côte d'Azur, Professor, HDR]

Post-Doctoral Fellows

- Stéphanie Challita [Inria, until Aug 2020]
- Jad Khatib [Inria, until Sep 2020]

PhD Students

- Joelle Abou Faysal [Renault, CIFRE]
- Carsten Bruns [Université Côte d'Azur]
- Joao Cambeiro [Université Côte d'Azur, from Oct 2020]
- Mansur Khazeev [Université Innopolis - Russie, from Nov 2020]
- Giovanni Liboni [Groupe SAFRAN, CIFRE]
- Hugo Pompougnac [Inria]
- Fabien Siron [Krono Safe, CIFRE, from Nov 2020]
- Enlin Zhu [CNRS, from Sep 2020]

Technical Staff

- Luc Hogie [CNRS, Engineer]
- Jad Khatib [Inria, Engineer, from Nov 2020]

Interns and Apprentices

- Sara El Khatab [Inria, from Jun 2020]
- Florian Hofhammer [Université Côte d'Azur, from Mar 2020 until Jul 2020]
- Abdul Qadir Khan [Inria, from Jun 2020]
- Oleksii Khramov [Université Côte d'Azur, from Oct 2020]
- Qian Liu [Inria, from Mar 2020 until Aug 2020]
- Rafael Mosca [Université Côte d'Azur, until Feb 2020]
- Biyang Wang [Université Côte d'Azur, from Mar 2020 until Aug 2020]

Administrative Assistant

- Patricia Riveill [Inria]

2 Overall objectives

The Kairos proposal ambitions to deal with the Design of Cyber-Physical Systems (CPS), at various stages, using Model-Based techniques and Formal Methods. Design here stands for co-modeling, co-simulation, formal verification and analysis activities, with connections both ways from models to code (synthesis and instrumentation for optimization). Formal analysis, in turn, concerns both functional and extra-functional correctness properties. Our goal is to link these design stages together, both vertically along the development cycle, and horizontally by considering the interactions between cyber/digital and physical models. These physical aspects comprise both physical environments and physical execution platform representations, which may become rather heterogeneous as in the cases of the Internet of Things (IoT) and computing at the edges of the gateways. The global resulting methodology can be tagged as Model-Based, Platform-Based CPS Design (Fig. 1).

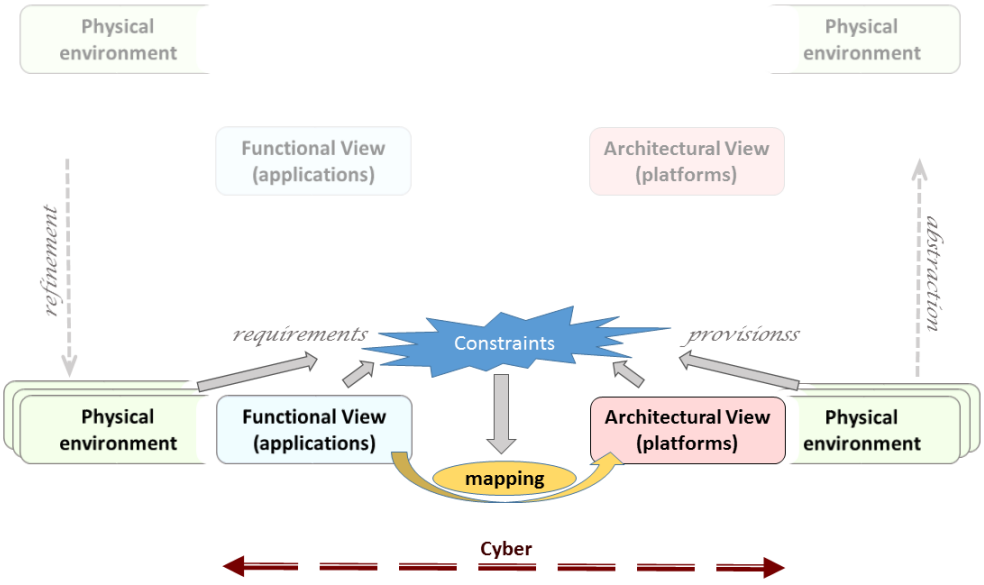

Figure 1: Cyber-Physical generic architectural features

CPS design must take into account all 3 aspects of application requirements, execution platform guarantees and contextual physical environment to establish both functional and temporal correctness. The general objective of Kairos is thus to contribute in the definition of a corresponding design methodology, based on formal Models of Computation for joint modeling of cyber and physical aspects, and using the important central concept of Logical Time for expressing the requirements and guarantees that define CPS constraints.

Logical Multiform Time. It may be useful to provide an introduction and motivation for the notion of Logical Multiform Time (and Logical Clocks), as they play a central role in our approach to Design. We call Logical Clock any repetitive sequence of occurrences of an event (disregarding possible values carried by the event). It can be regularly linked to physical time (periodic), but not necessarily so: fancy processors may change speeds, simulation engine change time-integration steps, or much more generally one may react with event-driven triggers of complex logical nature (do this after 3-times that unless this...). It is our belief that user specifications are generally expressed using such notions, with only partial timing correlations between distinct logical clocks, so that the process of realization (or "model-based compilation") consists for part in establishing (by analysis or abstract simulation) the possible tighter relations between those clocks (unifying them from a partial order of local total orders to a global total order). We have defined in the past a small language of primitives expressing recognized constraints structuring the relations between distinct logical clocks [1, 9]. This language (named CCSL for Clock Constraint Specification Language), borrows from notions of Synchronous Reactive Languages [12], Real-Time Scheduling Theory, and Concurrent Models of Computations and Communication (MoCCs) in Concurrency Theory [10] altogether. Corresponding extensions of Timed Models originally based on single (discrete or continuous) time can also be considered. Logical Time is used in our approach to express relation constraints between heterogeneous models, of cyber or physical origin, and to support analysis and co-simulation. Addressing cyber-physical systems demands to revisit logical time to deal with the multi-physical and sometimes uncertain environments.

In the following sections, we describe in turn the research agenda of Kairos on co-modeling, co-simulation, co-analysis and verification, and relation from models to code, respectively.

3 Research program

3.1 Cyber-Physical co-modeling

Cyber-Physical System modeling requires joint representation of digital/cyber controllers and natural physics environments. Heterogeneous modeling must then be articulated to support accurate (co-)simulation, (co-)analysis, and (co-)verification. The picture above sketches the overall design framework. It comprises functional requirements, to be met provided surrounding platform guarantees, in a contract approach. All relevant aspects are modeled with proper Domain Specific Languages (DSL), so that constraints can be gathered globally, then analyzed to build a mapping proposal with both a structural aspect (functions allocated to platform resources), but also a behavioral ones, scheduling activities. Mapping may be computed automatically or not, provably correct or not, obtained by static analytic methods or abstract execution. Physical phenomena (in a very broad acceptance of the term) are usually modeled using continuous-time models and differential equations. Then the "proper" discretization opportunities for numerical simulation form a large spectrum of mathematical engineering practices. This is not at all the domain of expertise of Kairos members, but it should not be a limitation as long as one can assume a number of properties from the discretized version. On the other hand, we do have a strong expertise on modeling of both embedded processing architectures and embedded software (i.e., the kind of usually concurrent, sometimes distributed software that reacts to and control the physical environment). This is important as, unlike in the "physical" areas where modeling is common-place, modeling of software and programs is far from mainstream in the Software Engineering community. These domains are also an area of computer science where modeling, and even formal modeling, of the real objects that are originally of discrete/cyber nature, takes some importance with formal Models of Computation and Communications. It seems therefore quite natural to combine physical and cyber modeling in a more global design approach (even multi-physic domains and systems of systems possibly, but always with software-intensive aspects involved). Our objective is certainly not to become experts

in physical modeling and/or simulation process, but to retain from it only the essential and important aspects to include them into System-Level Engineering design, based on Model-Driven approaches allowing formal analysis.

This sets an original research agenda: Model-Based System Engineering environments exist, at various stages of maturity and specificity, in the academic and industrial worlds. Formal Methods and Verification/Certification techniques also exist, but generally in a point-wise fashion. Our approach aims at raising the level of formality describing relevant features of existing individual models, so that formal methods can have a greater general impact on usual, "industrial-level", modeling practices. Meanwhile, the relevance of formal methods is enhanced as it now covers various aspects in a uniform setting (timeliness, energy budget, dependability, safety/security...).

New research directions on formal CPS design should focus on the introduction of uncertainty (stochastic models) in our particular framework, on relations between (logical) real-time and security, on relations between common programming languages paradigms and logical time, on extending logical frameworks with logical time, on the concern with resource discovery also in presence of mobility inherent to connected objects and Internet of Things [2, 8].

3.2 Cyber-Physical co-simulation

The FMI standard (Functional Mock-Up Interface) has been proposed for "purely physical" (i.e., based on persistent signals) co-simulation, and then adopted in over 100 industrial tools including frameworks such as Matlab/Simulink and Ansys, to mention two famous model editors. With the recent use of co-simulation to cyber-physical systems, dealing with the discrete and transient nature of cyber systems became mandatory. Together with other people from the community, we showed that FMI and other frameworks for co-simulation badly support co-simulation of cyber-physical systems; leading to bad accuracy and performances. More precisely, the way to interact with the different parts of the co-simulation require a specific knowledge about its internal semantics and the kind of data exposed (e.g., continuous, piecewise-constant). Towards a better co-simulation of cyber-physical systems, we are looking for conservative abstractions of the parts and formalisms that aim to describe the functional and temporal constraints that are required to bind several simulation models together.

3.3 Formal analysis and verification

Because the nature of our constraints is specific, we want to adjust verification methods to the goals and expressiveness of our modeling approach [13]. Quantitative (interval) timing conditions on physical models combined with (discrete) cyber modes suggest the use of SMT (Satisfiability Modulo Theories) automatic solvers, but the natural expressiveness requested (as for instance in our CCSL constructs) shows this is not always feasible. Either interactive proofs, or suboptimal solutions (essentially resulting of abstract run-time simulations) should be considered.

Complementarily to these approaches, we are experimenting with new variants of symbolic behavioural semantics, allowing to construct finite representations of the behaviour of CPS systems with explicit handling of data, time, or other non-functional aspects [4].

3.4 Relation between Model and Code

While models considered in Kairos can also be considered as executable specifications (through abstract simulation schemes), they can also lead to code synthesis and deployment. Conversely, code execution of smaller, elementary software components can lead to performance estimation enriching the models before global mapping optimization [3]. CPS introduce new challenging problems for code performance stability. Indeed, two additional factors for performance variability appear, which were not present in classical embedded systems: 1) variable and continuous data input from the physical world and 2) variable underlying hardware platform. For the first factor, CPS software must be analysed in conjunction with its data input coming from the physics, so the variability of the performance may come from the various data. For the second factor, the underlying hardware of the CPS may change during the time (new computing actors appear or disappear, some actors can be reconfigured during execution). The new challenge is to understand how these factors influence performance variability exactly, and how to

provide solutions to reduce it or to model it. The modeling of performance variability becomes a new input.

3.5 Code generation and optimization

A significant part of CPS design happens at model level, through activities such as model construction, analysis, or verification. However, in most cases the objective of the design process is implementation. We mostly consider the implementation problem in the context of embedded, real-time, or edge computing applications, which are subject to stringent performance, embedding, and safety *non-functional requirements*.

The implementation of such systems usually involves a mix of synthesis—(real-time) scheduling, code generation, compilation—and performance (*e.g.* timing) analysis. One key difficulty here is that synthesis and performance analysis depend on each other. As enumerating the various solutions is not possible for complexity reasons, heuristic implementation methods are needed in all cases. One popular solution here is to build the system first using unsafe performance estimations for its components, and then check system *schedulability* through a global analysis. Another solution is to use safe, over-approximated performance estimations and perform their mapping in a way that ensures by construction the schedulability of the system.

In both cases, the level of specification for the compound design space -including functional application, execution platform, extra-functional requirements, implementation representation- is a key problem. Another problem is the definition of scalable and efficient mapping methods based on both "exact" approaches (ILP/SMT/CP solving) and compilation-like heuristics.

3.6 Extending logical frameworks with logical time

The Curry-Howard isomorphism (*proposition-as-types and proofs-as-typed-λ-terms*) represent the logical and computational basis to interactive theorem provers: our challenge is to investigate and design time constraints within a Dependent Type Theory (e.g. if event A happened-before event B, then the timestamp/type of A is less (i.e. a subtype) than the timestamp/type of B). We are currently extending the Edinburgh Logical Framework (LF) of Harper-Honsell-Plotkin with relevant constructs expressing logical time and synchronization between processes. Also, union and intersection types with their subtyping constraints theories could capture some constraints expressions *à la* CCSL needed to formalize logical clocks (in particular CCSL expressions like subclock, clock union, intersection and concatenation) and provide opportunities for an *ad hoc* polymorphic timed Type Theory. Logical time constraints seen as property types can beneficially be handled by logical frameworks. The new challenge here is to demonstrate the relevance of Type Theory to work on logical and multiform timing constraint resolution.

3.7 Object-oriented programming and logical time

We formalized in the past object-oriented programming features and safe static type systems featuring delegation-based or trait inheritance: well-typed program will never produce the `message-not-found` infamous run-time error. We will view logical time as a mean to enhance the description of timing constraints and properties on top of existing language semantics. When considering general purpose object-oriented languages, like Java, Type Theory is a natural way to provide such kinds of properties. Currently, few languages have special types to manage instants, time structures and instant relations like subclocking, precedence, causality, equality, coincidence, exclusion, independence, etc. CCSL provides ad-hoc constructors to specify clock constraints and logical time: enriching object-oriented type theories with CCSL expressions could constitute an interesting research perspective towards a wider usage of CCSL. The new challenge is to consider logical time constraints as behavioral type properties, and the design of programming language constructs and *ad-hoc* type systems. Advances of typed-calculi featuring those static time features will be applied to our extension [7] of the lambda-calculus of objects of Fisher-Honsell-Mitchell.

3.8 Extensions for spatio-temporal modeling and mobile systems

While Time is clearly a primary ingredient in the proper design of CPS systems, in some cases Space, and related notions of local proximity or conversely long distance, play also a key role for correct modeling, often in part because of the constraints this puts on interactions and time for communications. Once space is taken into account, one has to recognize also that many systems will request to consider mobility, originated as change of location through time. Mobile CPSs (or mCPS) occur casually in real-life, e.g., in the case of Intelligent Transportation Systems, or in roaming connected objects of the IoT. Spatio-temporal and mobility modeling may each lead to dynamicity in the representation of constraints, with the creation/deletion/discovering of new components in the system. This opportunity for new expressiveness will certainly cause new needs in handling constraint systems and topological graph locations. The new challenge is to provide an algebraic support with a constraint description language that could be as simple and expressive as possible, and of use in the semantic annotations for mobile CPS design. We also aims to provide fully distributed routing protocols to manage Semantic Resource Discovery in IoT.

4 Application domains

4.1 Cyber-Physical and Embedded System design

System Engineering for CPS systems requires combinations of models, methods, and tools owing to multiple fields, software and system engineering methodologies as well as various digitalization of physical models (such as "Things", in Internet of Things (IoT)). Such methods and tools can be academic prototypes or industry-strength offers from tool vendors, and prominent companies are defining design flow usages around them. We have historical contacts with industrial and academic partners in the domains of avionics and embedded electronics (Airbus, Thales, Safran). We also have new collaborations in the fields of satellites (Thales Alenia Space) and connected cars driving autonomously (Renault Software Factory). These provide us with current use cases and new issues in CPS co-modeling and co-design (Digital Twins) further described in New Results section. The purpose here is to insert our formal methods into existing design flows, to augment their analysis power where and when possible.

4.2 Smart contracts for Connected Objects in the Internet Of Things

Due to increasing collaborations with local partners, we have recently considered Smart Contracts (as popularized in Blockchain frameworks), as a way to formally establish specification of behavioral system traces, applied to connected objects in a IoT environment. The new ANR project SIM is based on the definition of formal language to describe services for autonomous vehicles that would execute automatically based on the observation of what is happening on the vehicle or the driver. The key focus is on the design of a virtual passport for autonomous cars that would register the main events occuring on the car and would use them to operate automatic but trustworthy and reliable services.

4.3 Safe Driving Rules for Automated Driving

Self-driving cars will have to respect roughly the same safety-driving rules as currently observed by human drivers (and more). These rules may be expressed syntactically by temporal constraints (requirements and provisions) applied to the various meaningful events generated as vehicles interact with traffic signs, obstacles and other vehicles, distracted drivers and so on. We feel our formalisms based on Multiform Logical Time to be well suited to this aim, and follow this track in several collaborative projects with automotive industrial partners. This domain is a incentive to increase the expressiveness of our language and test the scalability of our analysis tools on real size data and scenari.

5 New software and platforms

5.1 New software

5.1.1 VerCors

Name: VERification of models for distributed communicating COmponants, with safety and Security

Keywords: Software Verification, Specification language, Model Checking

Functional Description: The VerCors tools include front-ends for specifying the architecture and behaviour of components in the form of UML diagrams. We translate these high-level specifications, into behavioural models in various formats, and we also transform these models using abstractions. In a final step, abstract models are translated into the input format for various verification toolsets. Currently we mainly use the various analysis modules of the CADP toolset.

Release Contributions: It includes integrated graphical editors for GCM component architecture descriptions, UML classes, interfaces, and state-machines. The user diagrams can be checked using the recently published validation rules from, then the corresponding GCM components can be executed using an automatic generation of the application ADL, and skeletons of Java files.

The experimental version (2019) also includes algorithms for computing the symbolic semantics of Open Systems, using symbolic methods based on the Z3 SMT engine.

News of the Year: The experimental version (2019) also includes: - algorithms for computing the symbolic semantics of Open Systems, using symbolic methods based on the Z3 SMT engine. - a stand alone textual editor for (open) pNet systems, that generates API code to construct their internal representation in the platform.

URL: https://team.inria.fr/scale/software/vercors/

Authors: Antonio Cansado, Eric Madelaine, Marcela Rivera, Ludovic Henrio, Oleksandra Kulankhina, Bartlomiej Szejna

Contact: Eric Madelaine

Participants: Antonio Cansado, Bartlomiej Szejna, Eric Madelaine, Ludovic Henrio, Marcela Rivera, Nassim Jibai, Oleksandra Kulankhina, Siqi Li, Xudong Qin, Zechen Hou

Partner: East China Normal University Shanghai (ECNU)

5.1.2 TimeSquare

Keywords: Profil MARTE, Embedded systems, UML, IDM

Scientific Description: TimeSquare offers six main functionalities:

- graphical and/or textual interactive specification of logical clocks and relative constraints between them,
- definition and handling of user-defined clock constraint libraries,
- automated simulation of concurrent behavior traces respecting such constraints, using a Boolean solver for consistent trace extraction,
- call-back mechanisms for the traceability of results (animation of models, display and interaction with waveform representations, generation of sequence diagrams...).
- compilation to pure java code to enable embedding in non eclipse applications or to be integrated as a time and concurrency solver within an existing tool.
- a generation of the whole state space of a specification (if finite of course) in order to enable model checking of temporal properties on it

Functional Description: TimeSquare is a software environment for the modeling and analysis of timing constraints in embedded systems. It relies specifically on the Time Model of the Marte UML profile, and more accurately on the associated Clock Constraint Specification Language (CCSL) for the expression of timing constraints.

URL: http://timesquare.inria.fr

Authors: Nicolas Chleq, Julien DeAntoni, Frédéric Mallet, Benoît Ferrero, Charles André

Contacts: Julien DeAntoni, Frédéric Mallet

Participants: Benoît Ferrero, Charles André, Frédéric Mallet, Julien DeAntoni, Nicolas Chleq

5.1.3 GEMOC Studio

Name: GEMOC Studio

Keywords: DSL, Language workbench, Model debugging

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

- Melange, a tool-supported meta-language to modularly define executable modeling languages with execution functions and data, and to extend (EMF-based) existing modeling languages.
- MoCCML, a tool-supported meta-language dedicated to the specification of a Model of Concurrency and Communication (MoCC) and its mapping to a specific abstract syntax and associated execution functions of a modeling language.
- GEL, a tool-supported meta-language dedicated to the specification of the protocol between the execution functions and the MoCC to support the feedback of the data as well as the callback of other expected execution functions.
- BCOoL, a tool-supported meta-language dedicated to the specification of language coordination patterns to automatically coordinates the execution of, possibly heterogeneous, models.
- Sirius Animator, an extension to the model editor designer Sirius to create graphical animators for executable modeling languages.

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

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

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

Contacts: Benoît Combemale, Julien DeAntoni

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

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

5.1.4 BCOol

Name: BCOoL

Keywords: DSL, Language workbench, Behavior modeling, Model debugging, Model animation

Functional Description: BCOoL is a tool-supported meta-language dedicated to the specification of language coordination patterns to automatically coordinate the execution of, possibly heterogeneous, models.

URL: http://www.gemoc.org

Authors: Julien DeAntoni, Matias Vara Larsen, Benoît Combemale, Didier Vojtisek

Contacts: Julien DeAntoni, Benoît Combemale

Participants: Julien DeAntoni, Matias Vara Larsen, Benoît Combemale, Didier Vojtisek

5.1.5 JMaxGraph

Keywords: Java, HPC, Graph algorithmics

Functional Description: JMaxGraph is a collection of techniques for the computation of large graphs on one single computer. The motivation for such a centralized computing platform originates in the constantly increasing efficiency of computers which now come with hundred gigabytes of RAM, tens of cores and fast drives. JMaxGraph implements a compact adjacency-table for the representation of the graph in memory. This data structure is designed to 1) be fed page by page, à-la GraphChi, 2) enable fast iteration, avoiding memory jumps as much as possible in order to benefit from hardware caches, 3) be tackled in parallel by multiple-threads. Also, JMaxGraph comes with a flexible and resilient batch-oriented middleware, which is suited to executing long computations on shared clusters. The first use-case of JMaxGraph allowed F. Giroire, T. Trolliet and S. Pérennes to count K2,2s, and various types of directed triangles in the Twitter graph of users (23G arcs, 400M vertices). The computation campaign took 4 days, using up to 400 cores in the NEF Inria cluster.

URL: http://www.i3s.unice.fr/~hogie/software/?name=jmaxgraph

Contacts: Luc Hogie, Michel Syska, Stéphane Pérennes

5.1.6 Lopht

Name: Logical to Physical Time Compiler

Keywords: Real time, Compilation

Scientific Description: The Lopht (Logical to Physical Time Compiler) has been designed as an implementation of the AAA methodology. Like SynDEx, Lopht relies on off-line allocation and scheduling techniques to allow real-time implementation of dataflow synchronous specifications onto multiprocessor systems. But there are several originality points: a stronger focus on efficiency, which results in the use of a compilation-like approach, a focus on novel target architectures (many-core chips and time-triggered embedded systems), and the possibility to handle multiple, complex non-functional requirements covering real-time (release dates and deadlines possibly different from period, major time frame, end-to-end flow constraints), ARINC 653 partitioning, the possibility to preempt or not each task, and finally SynDEx-like allocation.

Functional Description: Compilation of high-level embedded systems specifications into executable code for IMA/ARINC 653 avionics platforms. It ensures the functional and non-functional correctness of the generated code.

Authors: Dumitru Potop-Butucaru, Thomas Carle

Contact: Dumitru Potop-Butucaru

Participants: Dumitru Potop-Butucaru, Manel Djemal, Thomas Carle, Zhen Zhang

5.1.7 LoPhT-manycore

Name: Logical to Physical Time compiler for many cores

Keywords: Real time, Compilation, Task scheduling, Automatic parallelization

Scientific Description: Lopht is a system-level compiler for embedded systems, whose objective is to fully automate the implementation process for certain classes of embedded systems. Like in a classical compiler (e.g. gcc), its input is formed of two objects. The first is a program providing a platform-indepedent description of the functionality to implement and of the non-functional requirements it must satisfy (e.g. real-time, partitioning). This is provided under the form of a data-flow synchronous program annotated with non-functional requirements. The second is a description of the implementation platform, defining the topology of the platform, the capacity of its elements, and possibly platform-dependent requirements (e.g. allocation).

From these inputs, Lopht produces all the C code and configuration information needed to allow compilation and execution on the physical target platform. Implementations are correct by construction Resulting implementations are functionally correct and satisfy the non-functional requirements. Lopht-manycore is a version of Lopht targeting shared-memory many-core architectures.

The algorithmic core of Lopht-manycore is formed of timing analysis, allocation, scheduling, and code generation heuristics which rely on four fundamental choices. 1) A static (off-line) real-time scheduling approach where allocation and scheduling are represented using time tables (also known as scheduling or reservation tables). 2) Scalability, attained through the use of low-complexity heuristics for all synthesis and associated analysis steps. 3) Efficiency (of generated implementations) is attained through the use of precise representations of both functionality and the platform, which allow for fine-grain allocation of resources such as CPU, memory, and communication devices such as network-on-chip multiplexers. 4) Full automation, including that of the timing analysis phase.

The last point is characteristic to Lopht-manycore. Existing methods for schedulability analysis and real-time software synthesis assume the existence of a high-level timing characterization that hides much of the hardware complexity. For instance, a common hypothesis is that synchronization and interference costs are accounted for in the duration of computations. However, the high-level timing characterization is seldom (if ever) soundly derived from the properties of the platform and the program. In practice, large margins (e.g. 100\%) with little formal justification are added to computation durations to account for hidden hardware complexity. Lopht-manycore overcomes this limitation. Starting from the worst-case execution time (WCET) estimations of computation operations and from a precise and safe timing model of the platform, it maintains a precise timing accounting throughout the mapping process. To do this, timing accounting must take into account all details of allocation, scheduling, and code generation, which in turn must satisfy specific hypotheses.

Functional Description: Accepted input languages for functional specifications include dialects of Lustre such as Heptagon and Scade v4. To ensure the respect of real-time requirements, Lopht-manycore pilots the use of the worst-case execution time (WCET) analysis tool (ait from AbsInt). By doing this, and by using a precise timing model for the platform, Lopht-manycore eliminates the need to adjust the WCET values through the addition of margins to the WCET values that are usually both large and without formal safety guarantees. The output of Lopht-manycore is formed of all the multi-threaded C code and configuration information needed to allow compilation, linking/loading, and real-time execution on the target platform.

News of the Year: In the framework of the ITEA3 ASSUME project we have extended the Lopht-manycore to allow multiple cores to access the same memory bank at the same time. To do this, the timing

accounting of Lopht has been extended to take into account memory access interferences during the allocation and scheduling process. Lopht now also pilots the aiT static WCET analysis tool from AbsInt by generating the analysis scripts, thus ensuring the consistency between the hypotheses made by Lopht and the way timing analysis is performed by aiT. As a result, we are now able to synthesize code for the computing clusters of the Kalray MPPA256 platform. Lopht-manycore is evaluated on avionics case studies in the perspective of increasing its technology readiness level for this application class.

Authors: Keryan Didier, Dumitru Potop-Butucaru, Thomas Carle, Manel Djemal

Contact: Dumitru Potop-Butucaru

Participants: Dumitru Potop-Butucaru, Keryan Didier

5.1.8 Bull ITP

Name: The Bull Proof Assistant

Keywords: Proof, Certification, Formalisation

Scientific Description: Bull is a prototype theorem prover based on the Delta-Framework, i.e. a fully-typed Logical Framework à la Edinburgh LF decorated with union and intersection types, as described in previous papers by the authors. Bull is composed by a syntax, semantics, typing, subtyping, unification, refinement, and REPL. Bull also implements a union and intersection subtyping algorithm. Bull has a command-line interface where the user can declare axioms, terms, and perform computations and some basic terminal-style features like error pretty-printing, subexpressions highlighting, and file loading. Moreover, Bull can typecheck a proof or normalize it. These lambda-terms can be incomplete, therefore the Bull's typechecking algorithm uses high-order unification to try to construct the missing subterms. Bull uses the syntax of Berardi's Pure Type Systems to improve the compactness and the modularity of its Kernel. Abstract and concrete syntax are mostly aligned and similar to the concrete syntax of the Coq theorem prover. Bull uses a higher-order unification algorithm for terms, while typechecking and partial type inference are done by a bidirectional refinement algorithm, similar to the one found in Matita and Beluga. The refinement can be split into two parts: the essence refinement relative to the computational part and the typing refinement relative to its logical content. Binders are implemented using commonly-used de Bruijn indices. Bull comes with a concrete language syntax that will allow user to write Delta-terms. Bull also features reduction rules and an evaluator performing an applicative order strategy. Bull also feature a refiner which does partial typechecking and type reconstruction. Bull distribution comes with classical examples of the intersection and union literature, such as the ones formalized by Pfenning with his Refinement Types in LF and by Pierce. Bull prototype experiment, in a proof theoretical setting, forms of polymorphism alternatives to Girard's parametric one.

Functional Description: Bull is a prototype theorem prover based on the Delta-Framework, i.e. a fully-typed Logical Framework à la Edinburgh LF decorated with union and intersection types. Bull also implements a subtyping algorithm. Bull has a command-line interface where the user can declare axioms, terms, and perform computations and some basic terminal-style features like error pretty-printing, subexpressions highlighting, and file loading. Moreover, it can typecheck a proof or normalize it. Further extensions will include adding a tactic language, code extraction, and induction.

Release Contributions: First stable version

URL: https://github.com/cstolze/Bull/tree/master/bull

Authors: Claude Stolze, Luigi Liquori

Contacts: Claude Stolze, Luigi Liquori

Participants: Claude Stolze, Luigi Liquori

Partners: Inria, Université Paris-Diderot, Université d'Udine

5.1.9 CoSim20

Name: CoSim20: a Distributed Co-Simulation Environment

Keywords: Cosimulation, Development tool suite, Cyber-physical systems, Model-driven engineering

Functional Description: The Cosim20 IDE helps to build correct and efficient collaborative co-simulations. It is based on model coordination interfaces that provide the information required for coordination by abstracting the implementation details, ensuring intellectual property preservation. Based on such interfaces a system engineer can (graphically or textually) specify the intended communications between different simulation units. Then, the CoSim20 framework generates the artifacts allowing a distribution of the co-simulation where the simulation units communicate together at the required instants.

URL: https://gitlab.inria.fr/glose/model-coordination-language

Publications: hal-03038527, hal-02292048, hal-01675396, hal-03038547

Contact: Julien DeAntoni

Participants: Giovanni Liboni, Julien DeAntoni

Partner: SAFRAN tech

5.1.10 Idawi

Keywords: Java, Distributed, Distributed computing, Distributed Applications, Web Services, Parallel computing, Component models, Software Components, P2P, Dynamic components, Iot

Functional Description: Idawi is a middleware for the development and experimentation of distributed algorithms. It boasts a very general and flexible multi-hop component-oriented model that makes it applicable in many contexts such as parallel and distributed computing, cloud, Internet of Things (IOT), P2P networks. Idawi components can be deployed anywhere a SSH connection is possible. They exhibit services which communicate with each other via explicit messaging. Messages can be sent synchronously or asynchronously, and can be handled in either a procedural (with the optional use of futures) or reactive (event-driven) fashion. In the latter case, multi-threading is used to maximize both the speed and the number of components in the system. Idawi message transport is done via TCP, UDP, SSH or shared-memory.

Idawi is a synthesis of past developments of the COATI Research group in the field of graph algorithms for big graphs, and it is designed to be useful to the current and future Research project of COATI and KAIROS team-projects.

URL: https://github.com/lhogie/idawi

Contact: Luc Hogie

6 New results

6.1 Spatio-temporal constraints for mobile systems, with automotive driving assistance illustrations

Participants Frédéric Mallet, Joëlle Abou Faysal, Robert de Simone, Qian Liu.

The objective here is to extend constraint specifications to encompass spatial aspects in addition to logical multiform time. Spatio-temporal logics and requirement formalisms are thus an inspiration here. But mobility requests additionally that these spatio-temporal relations evolve in time. We are investigating in several directions:

- a target methodological approach is to consider these spatio-temporal relations to express safe driving rules as requirements or guarantees, meant to (in)validate trajectory proposals computed by a lower-level algorithmic system (itself operating on more direct neighborhood information). A realistic size case study is handled in collaboration with Renault Software Labs, as part of the CIFRE PhD contract of Joelle Abou-Faysal, to define the precise needs in expressiveness and formal validation.

- a pattern-based language combining features from spatio-temporal logics with flexibility of Multiple Logical Time to express temporal constraints based on logical events that are characteristic of Safe Driving Rules [27, 26]

- an extension of our previous work that consists in extending a spatio-temporal logics called STEC with logical time constructs. This logics is applied to smart transportation systems and the control of train systems. The spatial layout of train tracks are captured are the successive instants of a given logical clocks [16].

6.2 System Engineering for Performance and Availability in satellite embedded COTS

Participants Robert de Simone, Julien Deantoni.

In the context of the IRT ATIPPIC project, we applied Multiform Logical Time specifications to the case of Functional (and Temporal) Safety Analysis of satellite systems. At project completion, we presented its results in a dedicated forum [21].

6.3 Efficient solvers and provers for CCSL

Participants Frédéric Mallet, Min Zhang, Xiaohong Chen.

One of the goals of the team is to promote the use of logical time in various application domains. This requires to have efficient solvers for CCSL. We have made considerable progresses on this part along two lines. One by relying on SMT solvers (like Z3), the other by building a dynamic logic amenable to building semi-automatic proofs for logical time properties of reactive systems. Then for some classes of problems we can use efficient solving tools.

- The first step is to have an efficient Z3 library for solving CCSL specifications. We have improved a lot the performances over last year by getting rid of some of the existential quantifiers in our properties

- Second, we use this solver to help requirement engineers elicite the requirements [20]. We use execution traces to help generate valid satisfied CCSL specifications.

- Third, we have built a dynamic logics based on CCSL, where the formulae are derived from CCSL relational operators and programs include some of CCSL expressions and some imperative reactive constructs akin to Esterel programs [18, 17]. Then we have an interactive proof system, that helps prove that some reactive program satisfies a set of formulas at all time. As we use only a subset of CCSL then, we can restrict to a decidable subset of the logics and the SMT solver is always efficient. The SMT helps guide the semi-automatic proof by identifying the next proof rules that can be used (or not).

To measure the improvements of our solving algorithms we have considered two kinds of application cases:

- First, we capture the uncertainty in time specifications by replacing absolute dates by intervals. This greatly increases the possible state-space but we still managed to treat non trivial examples taken from the field of real-time scheduling [22],[29]. This extension of CCSL can be seen as a form of parametric logical time.
- Second, we consider large CCSL specifications that are automatically generated from a model (Sequence diagrams) applying a set of pre-defined patterns [20]. This systematic generation ends up in having large specifications that are used to stress our solver and better understand what makes it difficult to solve a CCSL specification.

6.4 Formalizing and extending Smart Contracts Languages with temporal and dynamic features

Participants Frédéric Mallet, Enlin Zhu, Ankica Baresic, Marie-Agnès Peraldi Frati, Robert de Simone, Luigi Liquori, Mansur Khazeev, Luis-Josef Amas.

"Smart Contracts", as a way to define legal ledger evolution in Blockchain environments, can be seen as rule constraints to be satisfied by the set of their participants. Such contracts are often reflecting requirements or guarantees extracted from a legal or financial corpus of rules, while this can be carried to other technical fields of expertise. Our view is that Smart Contracts are often relying on logically timed events, thus welcoming the specification style of our formalisms (such as CCSL). The specialization of multiform logical time constraints to this domain is under study, in collaboration with local academic partners at UCA UMRs LEAT and Gredeg, and industrial partners, such as Symag and Renault Software Labs. Local funding was obtained from UCA DS4H EUR, which allowed preparation of the ANR project SIM that was accepted in 2019. One goal is to get acceptance from the lawyers while still preserving strong semantics for verification or runtime verification [15].

Enlin Zhu and Ankica Baresic were recruited at the end of the year, respectively as PhD and postdoc students, to progress work in this area.

Mansur Khazeev was recruited as UCA PhD in September 2020, while funded by the University of Innopolis, Russia. His topic will focus on Modifying Smart Contract Languages (SCL), and more specifically on building multi-dimensional extensions to the SCLs landscape considering public and private blockchains.

The internship of Luis-Josef Amas explored the potentiality of the Solidity SCL to safely and dynamically change their behavior, and proposed improvements for enhanced flexibility, inspired from [31].

6.5 CCSL extension to Stochastic logical time

Participants Frédéric Mallet, Robert de Simone.

CCSL specifications allows distinct clocks with flexible, incompletely constrained inter-relations. One may think of probabilistic rates to provide stochastic variation limits, expressed in Multiform Logical Time. The objective is to provide constructs to introduce such relations for the inclusion/subclock and precedence/fasterThan partial orders, but to consider also combinators that associate them. Preliminary results have been obtained by Frédéric Mallet and Robert de Simone, in collaboration with fellow researchers from ECNU Shanghai. A main objective here is to work on minimal extensions of existing analysis frameworks and tools, rather relying on translation into complex verification environments in stochastic model-checking.

This task was one of the main goal of our associated team with ECNU. It was supposed to be treated with the help of two long-term visitors that could not come because of the sanitary situation. Instead, we made some interesting progress on a form of parametric extension of CCSL, where absolute dates are replaced by interval of possible values [22]. The stochastic extensions intend to better characterize those intervals by specifying discrete distributions.

6.6 Semantic Resource Discovery in IoT

Participants Luigi Liquori, Marie-Agnès Peraldi Frati, Sara El Khatab, Abdul Qadir Khan, Oleksii Khramov
.

Within the standards for M2M and the Internet of Things, managed by ETSI, oneM2M, and in the context of the ETSI 589 contract, we are looking for suitable mechanisms and protocols to perform a Semantic Resource Discovery as described in the previous Subsection. More precisely, we are extending the (actually weak) Semantic Discovery mechanism of the IoT oneM2M standard. The goal is to enable an easy and efficient discovery of information and a proper inter-networking with external source/consumers of information (e.g. a data bases in a smart city or in a firm), or to directly search information in the oneM2M system for big data purposes. oneM2M ETSI standard has currently rather weak native discovery capabilities that work properly only if the search is related to specific known sources of information (e.g. searching for the values of a known set of containers) or if the discovery is very well scoped and designed (e.g. the lights in a house). We submitted our vision in ETSI project submission "Semantic Discovery and Query in oneM2M" for extending oneM2M with a powerful Semantic Resource Discovery Service, taking into account additional constraints, such as topology, mobility (in space), intermittence (in time), scoping, routing etc. This proposal was taken up, and a consortium was formed that issued a standard proposal [37, 38].

6.7 Asynchronous Contact Tracing (ACT)

Participants Luigi Liquori, Enrico Scarrone, Suno Wood.

The papers [34] and [39] examine the use of IoT technology in contact tracing and introduces the concept of Asynchronous Contact Tracing (ACT). ACT identifies contacts with IoT connected objects that have been contaminated by the SARS-CoV-2 virus and works in synergy with solutions designed for manual contact tracing to identify and alert people who may have been infected by the virus. This shifts the paradigm from synchronously tracing the contacts of the people infected by Covid-19 to asynchronously tracing of contacts of materials (such as infected surfaces, waste water, air conditioning filters, etc.) that are hosting the SARS-CoV-2 virus. This enables people who have come into contact asynchronously with those particular materials to be alerted of a potential Covid-19 contagion, and, at the same time, it signals that one or more persons have been in contact with the material which is now spreading the SARS-CoV-2 virus. This process could be particularly effective, considering that the SARS-CoV-2 virus can survive for a significant time on certain materials. The level of contamination may on the nature of the surface and materials, the concentration of the virus, the ambient temperature, the season of the year, the level of humidity, and exposure to sun light. The period of contamination can span from a few hours to several days. The ACT process uses existing, ready-to-market IoT-based technology and well-established wireless network techniques. The process is not dependent on achieving a certain number of tests, or of people adopting it, in order for the results to be useful. Moreover, it does not require the transmission of any personal information by the user, thus respecting both EU GDPR and public sensibility to personal privacy. This process was inspired by Occam's Razor or the Law of Parsimony (Latin: Lex Parsimoniae), that states that entities and theories useful to solve a problem should not be multiplied unless necessary. On the contrary, simpler entities and theories are preferable to more complex ones because they are easier to test and more likely to be true.

6.8 Empirical study of Amdahl's law on multicore processors

Participants Carsten Bruns, Sid Touati.

Since many years, we have been observing a shift from classical multiprocessor systems to multicores, which tightly integrate multiple CPU cores on a single die or package. This shift does not modify the fundamentals of parallel programming, but makes harder the understanding and the tuning of the performances of parallel applications. Multicores technology leads to sharing of microarchitectural resources between the individual cores, which Abel et al. classified in storage and bandwidth resources. We empirically analyzed effects of such sharing on program performance, through repeatable experiments. We show that they can dominate scaling behavior, besides the effects described by Amdahl's law and synchronization or communication considerations. In addition, we view the physical temperature and power budget also as a shared resource. It is a very important factor for performance nowadays, since DVFS over a wide range is needed to meet these constraints in multicores. Furthermore, we demonstrated that resource sharing not just leads a flat speedup curve with increasing thread count but can even cause slowdowns. Last, we propose a formal modeling of the performances to allow deeper analysis. Our work aims to gain a better understanding of performance limiting factors in high performance multicores, it shall serve as basis to avoid them and to find solutions to tune the parallel applications.

6.9 Behavioral Equivalence of Open Systems

Participants Eric Madelaine, Biyang Wang.

We consider Open (concurrent) Systems where the environment is represented as a number of processes which behavior is unspecified. Defining their behavioral semantics and equivalences from a Model-Based Design perspective naturally implies model transformations. To be proven correct, they require equivalence of "Open" terms, in which some individual component models may be omitted. Such models take into account various kind of data parameters, including, but not limited to, time. The middle term goal is to build a formal framework, but also an effective tool set, for the compositional analysis of such programs.

We achieved this year our developments about the algorithms and implementation of FH-Bisimulation (strong version). This was presented in a POPL workshop, published in [23]. But strong bisimulation is often too strong a notion for comparing systems, and typically a high-level specification and its implementation would require a more involved notion, classicaly named "weak equivalence", that ignores the internal communications and synchronisations between the components of a system. For our parameterized open systems, we have defined a notion of weak bisimulation, and proved that under reasonable assumptions, it is preserved by composition of open systems, achieving a powerful compositional approach for their formal verification [30].

The corresponding development of algorithms and implementation on our VerCors platform, is in progress, in collaboration with ECNU Shanghai. A journal paper has been submitted in Jan. 2021.

6.10 Bull, A Type Checker for a Logical Framework with Union and Intersection Types

Participants Luigi Liquori, Claude Stolze.

The paper [28] presents the syntax, semantics, typing, subtyping, unification, refinement, and REPL of Bull, a prototype theorem prover based on the Δ-Framework i.e. a fully-typed Logical Framework à la Edinburgh LF decorated with union and intersection types, as described in previous papers by the authors. Bull also implements a subtyping algorithm for the Type Theory Ξ of Barbanera-Dezani-de'Liguoro. Bull has a command-line interface where the user can declare axioms, terms, and perform computations and some basic terminal-style features like error pretty-printing, subexpressions highlighting, and file loading. Moreover, it can typecheck a proof or normalize it. These terms can be incomplete, therefore the typechecking algorithm uses unification to try to construct the missing subterms. Bull uses the syntax of Berardi's Pure Type Systems to improve the compactness and the modularity of the kernel.

Abstract and concrete syntax are mostly aligned and similar to the concrete syntax of Coq. Bull uses a higher-order unification algorithm for terms, while typechecking and partial type inference are done by a bidirectional refinement algorithm, similar to the one found in Matita and Beluga. The refinement can be split into two parts: the essence refinement and the typing refinement. Binders are implemented using commonly-used de Bruijn indices. We have defined a concrete language syntax that will allow user to write Δ-terms. We have defined the reduction rules and an evaluator. We have implemented from scratch a refiner which does partial typechecking and type reconstruction. We have experimented Bull with classical examples of the intersection and union literature, such as the ones formalized by Pfenning with his Refinement Types in LF and by Pierce. We hope that this research vein could be useful to experiment, in a proof theoretical setting, forms of polymorphism alternatives to Girard's parametric one.

6.11 Co-Modeling for Better Co-Simulations

Participants Julien Deantoni, Giovanni Liboni.

A Collaborative simulation consists in coordinating the execution of heterogeneous models executed by different tools. In most of the approaches from the state of the art, the coordination is unaware of the behavioral semantics of the different models under execution; *i.e.*, each model and the tool to execute are seen as a black box. We highlighted that it introduces performance and accuracy problems [40].

In order to improve the performance and correctness of co-simulations, we proposed a language to define model behavioral interfaces, *i.e.*, to expose some information about the model behavioral semantics. We also proposed another language to make explicit the way to coordinate the different models by using dedicated connectors. The goal is to provide little information about the models to avoid intellectual property violations, but enough to allow an expert to make relevant choices concerning their coordination. This year, based on such description we proposed the *Cosim20* framework (`https://gitlab.inria.fr/glose/model-coordination-language`), which showed that it is possible to automatically generate a distributed coordination algorithm, which is more accurate and efficient than the one from the state of the art [25]. Additionally, as an enabler of the distributed coordination algorithm we defined a moldable and extensible API for the simulation units so that the semantic awareness is also true at runtime [24].

Finally, it is worth noticing that Giovanni Liboni wrote its PhD thesis that should be defended in the next months. In its document Giovanni presented a co-simulation that includes fault injection. This example is based on discussions with Safran about fault-tolerance modeling and also leverage results obtained in the former ATIPPIC project [21].

This work is realized in the context of the GLOSE project (see Section 7.1) in collaboration with Safran and other INRIA teams (namely HyCOMES and DiVerSE).

6.12 Abstraction in Co-Modeling and Co-Simulation

Participants Julien Deantoni, Joao Cambeiro, Frédéric Mallet.

Nowadays, system engineering is an activity where different engineering domains are put together to define a solution to a specific complex problem. The particularity of system engineering lies in the need for several distinct stakeholders, where each of them is developing a part of the whole system with their own dedicated languages and tools. For each stakeholder, there are different tools and languages depending on the development cycle. At each stage of the development, they develop more or less abstract models, giving more or less fidelity according to the final product. Of course an increase in the fidelity of the model usually implies an increase in the simulation time of this model. In this context, Model Based System Engineering is using simulation to understand the emerging behavior of the whole system. For that it is required to coordinate/federate the artifacts produced by each stakeholder; and consequently to do a collaborative simulation of all of them.

Simulating the whole system may require a lot of time and computing power. This drastically reduces the possibility to explore new solutions by using design space exploration. This also drastically reduces the possibility for one stakeholder to understand the impact of its solution on the other stakeholders.

In the PhD thesis of Joao Cambeiro started in October 2020, we are exploring different formalizations and frameworks to increase the agility and foster innovation in systems engineering. For that it should be possible to automatically use the appropriate level of fidelity for a specific model, according to the goal of the simulation; reducing then the simulation time of the whole system. Of course the selection of the fidelity level must be done cautiously since it must preserve the properties of interest. Also, there is a mandatory adaptation between models at different levels of fidelity since the interfaces and model of computations may be different. All this is under study, underlined by the notions of experimental frames, contracts, abstraction and refinements.

As part of Zhao Hui's PhD work, we have proposed a language to bring together subsets of existing predefined languages in a bid to combine their expressiveness. Rather than trying to build the ultimate unified language, sum of all languages, we would rather select meaningful features in existing languages and build a new language based on those features. This year, as an exercise we have used this approach in the domain of real-time scheduling [29].

6.13 Engineering and Debugging of Concurrency

Participants Julien Deantoni.

In the last year, in collaboration with Steffen Zschaler (King's college London) and the Diverse inria team, we experimented in the addition of different concurrent engines in the Gemoc studio. We figured out that it may be difficult for a user to debug concurrency since depending on the systems, some interleaving are not of interest and make difficult the exploration of significant ones.

From these experiments, we decided to built up upon the Debugger Adapter Protocol (https://microsoft.github.io/debug-adapter-protocol/) to allow concurrency debugging. It implies amongst other a place where *exploration strategies* can be defined by the user (the user being either the language designer or the system designer). The concurrency strategies hide the irrelevant interleaving to highlight the important ones. These strategies are defined independently of the execution engine and implemented into the Gemoc framework. A journal paper is under progress.

6.14 Formal Operational Semantics of Lingua Franca

Participants Julien Deantoni.

Lingua Franca (LF) is a polyglot coordination language for concurrent and possibly time-sensitive applications ranging from low-level embedded code to distributed cloud and edge applications (https://github.com/icyphy/lingua-franca/wiki/Overview). IT is developed by Edward Lee's group at University Berkeley. The language is promising but no formal semantics has been defined, yet.In order to define the tooled formal semantics of Lingua Franca, we defined the semantics in MoCCML (http://timesquare.inria.fr/moccml/). Currently, a first working version is provided and to be discussed with the Lingua Franca team (https://github.com/jdeantoni/LinguaFrancaInGemoc).

This exercise has different facets. 1) The challenge is to specify the formal semantics of a language that requires time jumps (instead of time tick enumeration like usually done in CCSL); 2) It brings visibility to the work done in the team; 3) It was the opportunity to develop the formal aspects of the GEMOC studio, and we now have generative techniques to automatically create an exhaustive exploration tool for any language developed using MoCCML in GEMOC studio. This exhaustive exploration encompasses the state of the concurrency model and the state of the runtime data, which can be abstracted by the user by overloading the *equals* operator. While still in a prototype form, results of the exploration can be imported

in the CADP tool for model checking, bringing model checking capabilities in any Domain-Specific Language written by following the GEMOC-MoCCML approach.

6.15 Efficient parallelism in shared memory

Participants Dumitru Potop Butucaru, Jad Khatib.

This work took place in 2 contexts: the PIA ES3CAP project, which funds the post-doc of Jad Khatib (see section 8.3, mainly in collaboration with Safran Aircraft Engines), and a point-to-point collaboration with IRT Saint-Exupéry.

In both cases, the objective was to advance our work on automatic parallelization of real-time applications to cover new multi-core architectures: the Kalray MPPA Coolidge many-core, classical multi-cores such with ARM and POWER architecture, and predictable embedded multi-cores such as the Infineon TC-27x and TC-3xx series.

On the Kalray MPPA Coolidge platform, we have defined an new memory access interference model. On the Infineon TC-27x platform, we have defined a parallelization strategy. On the POWER and ARM platforms we have already defined an efficient parallelization method. The method was evaluated, with very good results, on the T1042 quad-core processor, where it achieved 1.8x acceleration on 2 cores (and even improved the sequential code generation performance through careful memory allocation). We are currently writing several articles on the topic.

Based on this work, we have also presented a paper in ERTS2020 [19].

6.16 A Language and Compiler for High-Performance Real-Time Programming

Participant Hugo Pompougnac, Dumitru Potop Butucaru.

The Static Single Assignment (SSA) form has proven an extremely useful tool in the hands of compiler builders. First introduced as an intermediate representation (IR) meant to facilitate optimizations, it became a staple of optimizing compilers. More recently, its semantic properties[1] established it as a sound basis for High-Performance Computing (HPC) compilation frameworks such as MLIR (`https://mlir.llvm.org`), where different abstraction levels of the same application[2] share the structural and semantic principles of SSA, allowing them to co-exist while being subject to common analysis and optimization passes (in addition to specialized ones).

But while compilation frameworks such as MLIR concentrate the existing know-how in HPC compilation for virtually every execution platform, they lack a key ingredient needed in the high-performance embedded systems of the future—the ability to represent reactive control and real-time aspects of a system. They do not provide first-class representation and reasoning for systems with a cyclic execution model, synchronization with external time references (logical or physical), synchronization with other systems, tasks and I/O with multiple periods and execution modes.

And yet, while the standard SSA form does not cover these aspects, it shares strong structural and semantic ties with one of the main programming models for reactive real-time systems: dataflow synchrony, and its large and structured corpus of theory and practice of RTE systems design.

Relying on this syntactic and semantic proximity, we have extended the SSA-based MLIR framework to open it to synchronous reactive programming of real-time applications. We illustrated the expressiveness of our extension through the compilation of the pure dataflow core of the Lustre language. This allowed us to model and compile all data processing, computational and reactive control aspects of a signal processing application.[3] In the compilation of Lustre (as embedded in MLIR), following an initial

[1] Functional determinism while still allowing for limited concurrency.

[2] Ranging from ML dataflow graphs and linear algebra specifications down to affine loop nests and optimized (tiled, vectorized. . .) low-level code.

[3] A pitch tuning vocoder.

normalization phase, all data type verifications, buffer synthesis, and causality analysis can be handled using existing MLIR SSA algorithms. Only the initialization analysis specific to the synchronous model (a.k.a. clock calculus or analysis) requires specific handling during analysis and code generation phases, leading to significant code reuse.

The MLIR embedding of Lustre is non-trivial. As modularity based on function calls is no longer natural due to the cyclic execution model, we introduced a node instantiation mechanism. We also generalized the usage of the special undefined/absent value in SSA semantics and in low-level intermediate representations such as LLVM IR. We clarified its semantics and strongly linked it to the notion of absence and the related static analyses (clock calculi) of synchronous languages.

Our extension remains fully compatible with SSA analysis and code transformation algorithms. It allows giving semantics and an implementation to all correct SSA specifications. It also supports static analyses determining correctness from a synchronous semantics point of view.

First results in this direction, obtained in collaboration with Google, are presented in [35].

6.17 Formal verification of logical execution time (LET) applications

Participants Robert de Simone, Dumitru Potop Butucaru, Fabien Siron.

Logical Execution Time (LET) is a paradigm of real-time systems design. It is based on the introduction of abstract/logical time bases (*e.g.* microsecond, engine cycle, *etc.*) allowing the natural specification of the activation *and duration* of the various operations/tasks of a cyber-physical system. The specification and high-level analysis of the system, as well as resource allocation (*e.g.* construction of scheduling tables) can be done at this abstract level. However, these time bases are logical: their relation with the (multiple) physical time measures (*e.g.* CPU clock cycles, rotation sensors) must be formally defined and proved correct separately.

In this context, we have started a collaboration with the Krono-Safe SME (`https://www.krono-safe.com`) aiming at:

- Verifying the correctness of a static scheduling plan with respect to a logical execution time specification provided in the PsyC language designed and developed by Krono-Safe.
- Validating a logical time specification provided in PsyC with respect to logical time safety properties.

As part of this collaboration, the CIFRE PhD of Fabien Siron started in 2020.

6.18 Understanding microarchitectural effects on the performance of parallel applications

Participants Carsten Bruns, Sid Touati.

Since several years, classical multiprocessor systems have evolved to multicores, which tightly integrate multiple CPU cores on a single die or package. This technological shift leads to sharing of microarchitectural resources between the individual cores, which has direct implications on the performance of parallel applications. It consequently makes understanding and tuning these significantly harder, besides the already complex issues of parallel programming. In this work, we empirically analyze various microarchitectural effects on the performance of parallel applications, through repeatable experiments. We show their importance, besides the effects described by Amdahl's law and synchronization or communication considerations. In addition to the classification of shared resources into storage and bandwidth resources of Abel et al., we view the physical temperature and power budget also as a shared resource. This is a very important factor for performance nowadays, since DVFS over a wide range is needed to meet the constraints in multicores. Our work aims to gain a better understanding of performance limiting factors in high performance multicores, it shall serve as a basis to avoid them and to find solutions to tune parallel applications. This work is submitted for publication (under review process).

6.19 Stack Buffer Overflows: Attacks and defense mechanisms

Participants Florian Hofhammer, Sid Touati.

This work is a master internship of Florian Hofhammer from École Polytechnique (Institut Polytechnique de Paris). The goal here is to summarize different stack buffer overflow attack and defense methods in a single document. In this context, the question arises how some defense methods can be bypassed by an attacker and how a defender can improve on the defense mechanisms to thwart stack buffer overflow based attacks. To achieve these goals and answer these questions, several attack methods and their implications on the security of currently deployed defense mechanisms are presented. They show that even on modern systems, stack buffer overflow vulnerabilities can be exploited to execute arbitrary code controlled by an attacker under specific circum- stances. Although stack buffer overflows are hard or impossible to exploit depending on the circumstances of the vulnerability and the system configuration, the need for security measure enhancements arises. Such improvements and expansions on current defense mechanisms against stack buffer overflows are also presented and compared with regard to their performance implications on a system.

6.20 Trustworthy Fleet Deployment and Management

Participants Nicolas Ferry.

This activity is starting as a result of Nicolas Ferry joining the team and as a continuation of his activities within the ENACT H2020 project. Continuous and automatic software deployment is still an open question for IoT systems, especially at the Edge and IoT ends. The state-of-the-art Infrastructure as Code (IaC) solutions are established on a clear specification about which part of the software goes to which types of resources. This is based on the assumption that, in the Cloud, one can always obtain the exact computing resources as required. However, this assumption is not valid on the Edge and IoT levels. In production, IoT systems typically contains hundreds or thousands of heterogeneous and distributed devices (also known as a *fleet of IoT/Edge devices*), each of which has a unique context, and whose connectivity and quality are not always guaranteed. In ENACT we both investigated the challenge of automating the deployment of software on heterogeneous devices and of managing variants of the software which fit different types or contexts of Edge and IoT devices in the fleet. The natural next step is to investigate how to guarantee the trustworthiness of the deployment when (i) the quality of the devices is not guaranteed, (ii) the context of each devices is continuously changing in an unanticipated manner, and (iii) software components are frequently evolving in the whole software stack of each device. In such context, ensuring the proper ordering and synchronization of the deployment actions is critical improve quality, trustworthiness and minimize downtime.

7 Bilateral contracts and grants with industry

7.1 Bilateral contracts with industry

Safran : Desir/Glose We participate to the bilateral collaborative program Desir, put up by Safran to work with selected academic partners. We share the Glose project started in this program with two other Inria teams : HyComes, and DiverSE. The aim of the project is to improve early stages of system engineering by allowing early execution and co-simulation of heterogeneous models. The technical content of our contributions is described in section 6.11. A CIFRE PhD is funded by Renault Software Factory on related topics.

Airbus In the continuation of the ITEA3 ASSUME project, Airbus has provided funding for the extension of the Real-Time Systems Compilation method to allow parallelization onto multi-cores with

classical ARM or POWER architecture. The technical content of our contributions is described in section 6.15. The technical content of our contributions is described in section 6.2.

IRT Saint-Exupery The CAPHCA project of IRT Saint-Exupéry has provided funding for the extension of the Real-Time Systems compilation method to allow parallelization onto timing predictable multi-cores different from the Kalray MPPA 256. The targets of this work are Infineon TC27x and FlexPRET.
WE are due to start a new collaborative project in similar context, named ARCHEOCS, on the development of formal development methods for optimized mapping (allocation + scheduling) in embedded system engineering flows.

Renault Software Labs We have started, at the end of 2018, a collaboration with Renault Software Labs on the definition of rules for ensuring safe maneuvers in autonomous vehicles. The rules express conditions from the environments, safety rules to preserve the integrity of the vehicles, driving legislation rules, local rules from the authorities. The rules must be updated dynamically when the vehicle evolves and are used to monitor at run-time the behavior of the ADAS. While the ADAS contains several algorithms relying on machine learning, the monitoring system must be predictive and rules must guarantee formally that the system does not cause any accident. So it can be seen as a way to build trustworthy monitoring of learning algorithms. A CIFRE PhD is funded by Renault on this topic and has started in April 2019.

7.2 Bilateral grants with industry

ETSI standardization We received support to conduct the specification and proof-of-concept of Advanced Semantic Resource Discovery standard during the course of its development and adoption. The activity proposed is the study and development of semantic Discovery and Query capabilities for oneM2M and its contribution to the oneM2M standard.

The goal is to enable an easy and efficient discovery of information and a proper interworking with external source/consumers of information (e.g. a distributed data base in a smart city or in a firm), or to directly search information in the oneM2M system for big data purposes.

The STF is funded by ETSI and integrated within the Work Programme of the Technical Committee SmartM2M (TC SmartM2M).

Apart from the task related to the STF project management, the STF works on the development of the deliverables of the STF (i.e. Technical Reports, Progress Reports and contributions to oneM2M) within the following tasks:

T1 Discovery and Query use cases and requirements

T2 Discovery and Query options analysis and selection

T3 Discovery and Query simulation and evaluation

T4 Discovery and Query solution development

T5 Contribution to oneM2M.

8 Partnerships and cooperations

8.1 International initiatives

8.1.1 Inria International Labs

PLoT4IoT

Title: *Probabilistic Logical Time for Internet of Things*

Duration: 2020 - 2023

Coordinator: Frederic Mallet

Partners:

- School of Computer Science and Software Engineering, ECNU (China)

Inria contact: Frederic Mallet

Summary: The growing importance of Connected Objects in the Internet of Things (IoT) poses new challenges concerning modeling and design of so-called Cyber-Physical Systems (CPS), where cyber/discrete controler programs interplay with physical (often continuous) environments. While there are generally well-established modeling practices in physical science domains (often including discretization), the need for equally formal modeling treatment of reactive control software itself becomes all the more important, since correctness of functional and non-functional properties relies on the whole range of models, including the software executable specification models. The scope of the PLoT4IoT proposal is entirely devoted to the definition and analysis of modeling paradigms relevant for design of (mainly the cyber part of) CPS.

In this context, we shall study extensions to Logical Time models for CPS, dedicated one one hand to uncertainty and variability, on the other hand to spatio-temporal aspects and mobility. These models will be proposed for standardisation into the forthcomming version of the OMG UML MARTE profile. On a more abstract level, we shall study the semantics and the analysis methods for open (concurrent) systems featuring explicit handling of data, time, locations, and propose new approaches to formal verification of safety properties of these systems. As a transversal mathematical tool for these works, we plan to develop new efficient stategies for "Satisfiability Modulo Theory" tools adapted to the 3 three modeling theories above.

8.2 European initiatives

8.2.1 FP7 & H2020 Projects

H2020 EU Project ENACT The ENACT project is funded the European Union's H2020 Programme under grant agreement no 78035. ENACT started in 2018 and is planned to end in April 2021. Nicolas Ferry is the technical manager of ENACT and, while moving from SINTEF to Kairos, he will keep is role until the end of the project. The project consortium is made up of 12 partners from six countries and is coordinated by SINTEF. The overall goal of ENACT is to develop novel solutions to enable the DevOps of trustworthy smart IoT systems (IoT systems involving both sensors and actuators). More precisely, Nicolas continues his contributions to two research activities in ENACT: (i) solution for the trustworthy deployment of IoT systems (in collaboration with SINTEF) and (ii) the development of solutions to manage actuation conflicts with logical and temporal properties (in collaboration with Sparks team from i3s laboratory).

8.3 National initiatives

ANR Project SIM The ANR SIM (Smart IoT for Mobility) is a PRCE project co-funded by ANR (AAPG 2019) and DGA for 42 months. The national coordinator is the LEAT (UMR CNRS) and the other partners are Renault Software Labs and Symag. The goal is to provide a formal meta-language to describe smart contracts that can be used in the context of an autonomous vehicles to provide services to the users. The services are related to the combined use of multi-model transportation systems by having a single smart contracts that can enforce all the intermediate transactions with all the actors involved (car manufacturing, parking lease, highway toll companies, insurances, bike rental companies).

Competitivity Clusters The Kairos team is involved in the actions of the cluster SCS (Systèmes Communicants Sécurisés) and Frédéric MALLET is elected in the steering committee of SCS. One of the most prominent actions is to build, in partnership with Aix-Marseille University, a Digital Innovation Hub, to open the access (with actions of transfer and valorization) to Digital Innovations for companies that would benefit from it, like public institutions (hospitals, human resources, employment institutions) or private companies that could use IoT for agriculture, tourism, smart infrastructures (harbours, buildings, cities).

CNRS GDRs We are registered members of three GDR funded by CNRS : **SoC**^{2}, on topics of Hardware-software codesign and Non-Functional Property modeling for co-simulation; **LTP**, on verification and language design for reactive CPS systems; **GPL**, on software engineering and Domain-Specific Languages.

Inria Project Lab SPAI This collaborative action, targeting *Security by Program Analysis for the IoT (SPAI)*, is headed by the Indes Project, and associated the Antique, Privatics and Celtique EPIs.

PAI ES3CAP ES3CAP (Embedded Smart Safe Secure Computing Autonomous Platform) is a PIA (Programme d'Investissements d'Avenir) project. Its budget is of 22.2MEuros, over 36 months. The national coordinator is Kalray, and other partners include Safran, Renault, and MBDA. The objectives of the project are to:

- Build a hardware and software industry-grade solution for the development of computation-intensive critical application. The solution should cover the needs of industrial end users, and target multi/many-core hardware platforms. The solution will come with 3 to 6 usage profiles specific to various industries (automotive, aerospace, defence)
- Improve the technology readiness level of the proposed development flow from TRL4-5 (technology development) to TRL6-7, thus approaching as much as possible commercialization.
- Build an alternate, perennial ecosystem for critical real-time OSs and development tools for computer vision, data fusion and neural networks. The tools and components must be available on a prototyping and demonstration platform that is safe and secure.
- Capitalize on the convergence between the automotive and aerospace markets on subjects such as security, safety, decision making, and big data.

Our technical contributions to this project are described in 6.15. This project partially finances Hugo Pompugnac's PhD and Jad Khatib's post-doc.

8.4 Regional initiatives

PSPC ADAVEC Partners of this regional PACA projects are companies AVISTO, Epic&poc, and informally Renault Software Factory. We appear as UCA partner. The purpose of the project is to study handover procedures between Automated and/or Human Driving (under loss of ability on either side). Our contribution is to provide temporal safety rules for detection of relevant events in use cases.

9 Dissemination

9.1 Promoting scientific activities

9.1.1 Scientific events: organisation

ETSI oneM2M Webminar "Advanced Semantic Discovery" Décember 2020.

General chair, scientific chair Julien Deantoni is general chair of the 24th IEEE Forum on specification and Design Language (FDL 2021: `http://fdl-conference.org/`)

Member of the organizing committees Nicolas Ferry is member of the MDE4IoT workshop 2021 (co-located with ECMFA) organization committee.

Julien Deantoni was member of the MPM4CPS workshop 2020 (co-located with Models) organization committee.

9.1.2 Scientific events: selection

Chair of conference program committees Julien Deantoni was TPC co-chair of the 23rd IEEE Forum on specification and Design Languages (FDL 2020).

Member of the conference program committees in 2020, Julien Deantoni was TPC member of the following conferences and workshops: Modeling Language Engineering and Execution, Multi Paradigm Modeling for Cyber Physical Systems, Formal Co-Simulation of Cyber-Physical Systems, Formal Approaches for Advanced Computing Systems, specification and Design Language, Summer Simulation Conference.

Nicolas Ferry is a TPC member of the forthcoming conference and workshop: SEAA'2021 conference (Euromicro Conference on Software Engineering and Advanced Applications); First SWForum Workshop on Trustworthy Software and Open Source.

Robert de Simone was TPC member for EmSoft, DATE, FDL, and MemoCode 2020 conference editions.

Frédéric Mallet was TPC member of TASE, FDL, ModelsWard 2020 conference editions.

9.1.3 Journal

Member of the editorial boards Nicolas Ferry is one of the editors of the special issue on Model-Driven Engineering for the Internet of Things (MDE4IoT) of the System and Software Modeling journal.

Frederic Mallet was a guest editor for one special issue of Elsevier's Science of Computer Programming Journal [14]. He is also a member of the Editorial Board of the Journal of Software-Intensive Cyber-Physical Systems (SICS) from Springer.

Reviewer - reviewing activities In 2020, Julien Deantoni has been reviewer for the followings journals: ACM Transaction on Embedded Computing Systems, Springer System and Software Modeling.

In 2020, Luigi Liquori has been reviewer of the International conference FOSSACS 2020 and the international journals Theoretical Computer Science, Journal of Logical and Algebraic Methods in Programming, and Internet of Things Journal.

In 2020, Frédéric Mallet was reviewer for ACM Transaction on Embeded Computing Systems (TECS), ACM Transactions on Internet Technology (TOIT), Springer Software and Systems Modeling (SoSyM) Journal and Elsevier's Science of Computer Programming.

9.1.4 Invited talks

Luigi Liquori was invited to give a speech on "Asynchronous Contact Tracing (ACT)", heading the session on *Interoperability - Holistic COVID-19 tracing practices*, at the Knowledge Society Forum 2020 (online meeting), http://members.eurocities.eu/eurocities/calendar/events_list/Knowledge-Society-Forum-2020-WSPO-BSDCCF, September 2020.

9.1.5 Leadership within the scientific community

Robert de Simone is Steering Commitee member of the EmSoft international conference.

Luigi Liquori is Deleguee in the ETSI/OneM2M oneM2M - Standards for M2M and the Internet of Things.

Frederic Mallet was Topic Chair of Topic D1 on System Modelling and Specification for DATE Conference 2019, 2020 and 2021.

9.1.6 Scientific expertise

Nicolas Ferry is in the external advisory board of the SODALITE H2020 project (https://www.sodalite.eu).

Frédéric Mallet was an external expert for two CIFRE Projects submitted to ANRT.

Frédéric Mallet is the scientific advisor for UCA President regarding the Move2Digital answer to the call on European Digital Innovation Hubs (eDIHs) for Region Sud.

9.1.7 Research administration

Frédéric Mallet is the Deputy Director of I3S Laboratory (UMR CNRS 7271) with around 120 permanent researchers and faculties.

Frédéric Mallet is an elected member of the National Council of Universities (CNU) in Section 27 (during 4 years).

Frédéric Mallet is a member of the steering committee of EUR DS4H focusing on projects around Digital Systems for Humans.

Julien Deantoni is principal investigator of the bilateral GLOSE project in collaboration with Safran.

Robert de Simone was Inria representative for the DESIR joint research programme between Inria and Safran.

9.2 Teaching - Supervision - Juries

9.2.1 Teaching

- Licence : Sid TOUATI, Fondement machine, 75 heures eq TD, L1 informatique, Université Côte d'Azur.
- Licence : Sid TOUATI, Architecture machine, 45 heures eq TD, L3 informatique, Université Côte d'Azur.
- Licence : Sid TOUATI, Compilation, 33 heures eq TD, L3 informatique, Université Côte d'Azur.
- Master: Sid TOUATI, Architectures et logiciels hautes performances, 81 heures eq TD, Master 1 informatique, Université Côte d'Azur.
- Master international: Sid TOUATI, Advanced operating systems, 30 heures eq TD, Master 1 informatique, Université Côte d'Azur.
- International Master: Frédéric Mallet, Safety-Critical Systems, 32h, M1, Université Côte d'Azur.
- International Master: Frédéric Mallet, Software Engineering, 32h, M1, Université Côte d'Azur.
- Master: Frédéric Mallet, Programmation Synchrone, 32h, M1, Université Côte d'Azur.
- Master: Robert de Simone, Formal Methods for NoC-based design, 36 heures eq TD, M2 International Ubinet, Université Côte d'Azur.
- Licence: Marie-Agnès Peraldi-Frati, Web security (20h eq TD), Security of connected objects (20h eq TD), IoT Infrastructure deployment (20 H) and Large scale plateform for IoT (20h eq TD), in a licence cursus dedicated to Internet of Objects, Infrastructure and Applications, Université Côte d'Azur.
- Master: Marie-Agnès Peraldi-Frati, Web Security and IoT Platform, 20h eq TD, Master 2 SICOM, Univ Avignon.
- Master: Luigi Liquori, Peer-to-peer systems, 32 eq TD, Université Côte d'Azur.
- Master: Julien Deantoni, Finite State Machine, 54h eq TD, Polytech'Nice.
- Master: Julien Deantoni, Multi Paradigm Programming in C++, 54h eq TD, Polytech'Nice.
- Master: Julien Deantoni, Domain Specific Languages, 32h eq TD, Polytech'Nice.
- Master: Julien Deantoni, Language Interpreter, 32h eq TD, Polytech'Nice.
- Master: Julien Deantoni, Micro-controller programming, 8h eq TD, Polytech'Nice.
- Master: Dumitru Potop-Butucaru, A synchronous approach to the design of embedded real-time systems, 30h, EPITA Engineering School, Paris.
- Master: Dumitru Potop-Butucaru, Real-time embedded systems, 42h, EIDD (École d'Ingenieur Denis Diderot), Paris.

- Licence: Nicolas Ferry, Web programming for mobile devices (50h eq TD), Project Management and continuous delivery (25h eq TD), in a licence cursus dedicated to the development of mobile applications, Université Côte d'Azur.
- Licence: Nicolas Ferry, Software architecture (25h eq TD) and Programming Web Services (20h eq TD), in a licence cursus dedicated to Internet of Objects, Infrastructure and Applications, Université Côte d'Azur.
- License: Luc Hogie, Distributed programming, 28h eq TD, DUT Informatique.

9.2.2 Supervision

- PhD in progress: Carsten Bruns, Performance analysis and optimisation of C++ applications, Université Côte d'Azur, 2021, Sid TOUATI.
- PhD in progress: Giovanni Liboni, Coordination of discrete (Cyber) Models, Université Cote d'Azur, end 2021, Frédéric Mallet, Julien DeAntoni.
- PhD in progress: Joelle Abou Faysal, A Formal Language for Ensuring Safety Scenarios in Autonomous Vehicules, Université Cote d'Azur, 2022, Frédéric Mallet.
- PhD: Keryan Didier, Contributions to the safe and efficient parallelisation of hard real-time systems. Sorbonne University, September 19, 2019, Dumitru Potop-Butucaru.
- PhD in progress: Hugo Pompougnac, Sorbonne University, 2022, Dumitru Potop-Butucaru.
- PhD in progress: Fabien Siron, Méthodologie de vérification formelle de propriétés temporelles pour les applications temps-réel critique, Université Cote d'Azur, end 2023, Dumitru Potop-Butucaru.
- PhD in progress: Joao Cambeiro, Seamless Integration of Heterogeneous Multi Fidelity Models into a Systems Engineering Approach, Université Cote d'Azur, end 2023, Julien DeAntoni.
- PhD in progress: Enlin Zhu, Formal Language for Legally-Binding Smart Contracts, Université Cote d'Azur, end 2023, Frédéric Mallet.
- PhD in progress 2020-2023. Mansur Khazeev, *Self-amending Programming Languages for Dynamic Smart Contracts in Cryptocurrency*, Université Cote d'Azur, end 2023, Luigi Liquori.
- Ph.D. (waiting for VISA) 2020-2023. Laïka Binte Imran, *Resource Discovery in the Internet of Things*, Université Cote d'Azur, end 2023, Luigi Liquori.

9.2.3 Juries

- Julien Deantoni was reviewer for the PhD Jury of Saloua Bennani (Université de Toulouse, Jean Jaurès), July 7th 2020.
- Julien Deantoni was reviewer for the PhD Jury of Renan Leroux (Université de Toulouse, Paul Sabatier), October 1st 2020.
- Julien Deantoni was a member of the PhD Jury of Valentin Besnard (ENSTA Bretagne), December 9th 2020.
- Luigi Liquori was reviewer of the Ph.D. Jury of the Ph.D. of Mohamed Emine Laarouchi, CEA-Télécom Sud-Paris, 2020.
- Frédéric Mallet was a reviewer for the PhD jury of Mathieu Montin from Toulouse INP, September, 14th 2020.

9.2.4 Teaching Administration

Since March 2020, Frédéric Mallet is the director of the new department of informatics that brings together all three departments on Computer Sciences from Université Côte d'Azur gathering around 90 faculty staff from EUR DS4H, the Engineering School Polytech and the Technological Institute (IUT).

9.3 Popularization

9.3.1 Standardization

Luigi Liquori took active part in the following standardization events:

- "ETSI/STF 589: TR 103 715 SmartM2M; Study for oneM2M; Discovery and Query solutions Analysis & Selection", SmartM2M Plenary, September, 2020.
- "ETSI/STF 589: Advanced Semantic Discovery", SmartM2M/oneM2M WG2 System Design & Security, July 2020.
- Asynchronous Contact Tracing System; Fighting pandemic disease with Internet of Things, SmartM2M plenary meeting, June 2020.
- "Can IoT participate in Contact Tracing?", ETSI/eHEALTH-ATTM-SDMC Ad Hoc meeting on COVID-19 collaboration, June 2020.
- "ETSI/STF 589: Semantic Discovery in Presence of a "network" of M2M Service Providers" SmartM2M/oneM2M WG1 Requirements & Domain Models, April 2020.

9.3.2 Interventions

Julien Deantoni is part of the *"cordées de la réussite"* programme, where he does appearances in the Emile Roux school (le Cannet) to give 12 year-old kids courses on robotics .

10 Scientific production

10.1 Major publications

[1] C. André, J. Deantoni, F. Mallet and R. De Simone. 'The Time Model of Logical Clocks available in the OMG MARTE profile'. In: *Synthesis of Embedded Software: Frameworks and Methodologies for Correctness by Construction*. Ed. by S. K. Shukla and J.-P. Talpin. Chapter 7. Springer Science+Business Media, LLC 2010, July 2010, p. 28. URL: https://hal.inria.fr/inria-00495664.

[2] Y. Bao, M. Chen, Q. Zhu, T. Wei, F. Mallet and T. Zhou. 'Quantitative Performance Evaluation of Uncertainty-Aware Hybrid AADL Designs Using Statistical Model Checking'. In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 36.12 (Dec. 2017), pp. 1989–2002. DOI: 10.1109/TCAD.2017.2681076. URL: https://hal.inria.fr/hal-01644285.

[3] T. Carle, D. Potop-Butucaru, Y. Sorel and D. Lesens. 'From Dataflow Specification to Multiprocessor Partitioned Time-triggered Real-time Implementation *'. In: *Leibniz Transactions on Embedded Systems* (Nov. 2015). DOI: 10.4230/LITES-v002-i002-a001. URL: https://hal.inria.fr/hal-01263994.

[4] L. Henrio, E. Madelaine and M. Zhang. 'A Theory for the Composition of Concurrent Processes'. In: *36th International Conference on Formal Techniques for Distributed Objects, Components, and Systems (FORTE)*. Ed. by E. Albert and I. Lanese. Vol. LNCS-9688. Formal Techniques for Distributed Objects, Components, and Systems. Heraklion, Greece, 2016, pp. 175–194. DOI: 10.1007/978-3-319-39570-8_12. URL: https://hal.inria.fr/hal-01432917.

[5] F. Honsell, L. Liquori, P. Maksimovic and I. Scagnetto. 'LLFP : A Logical Framework for modeling External Evidence, Side Conditions, and Proof Irrelevance using Monads'. In: *Logical Methods in Computer Science*. Special Issue in honor of Pierre Louis Curien (Feb. 2017). URL: https://hal.inria.fr/hal-01146059.

[6] F. Jebali and D. P. Butucaru. 'Ensuring Consistency between Cycle-Accurate and Instruction Set Simulators'. In: *18th International Conference on Application of Concurrency to System Design, ACSD 2018, Bratislava, Slovakia, June 25-29, 2018*. 2018, pp. 105–114. DOI: 10.1109/ACSD.2018.00019. URL: https://doi.ieeecomputersociety.org/10.1109/ACSD.2018.00019.

[7] L. Liquori and C. Stolze. 'The Delta-calculus: syntax and types'. In: *FSCD 2019 - 4th International Conference on Formal Structures for Computation and Deduction*. 2019-06-24 and 2019-06-24. Dortmund, Germany, June 2019. URL: https://hal.archives-ouvertes.fr/hal-01963662.

[8] L. Liquori, C. Tedeschi, L. Vanni, F. Bongiovanni, V. Ciancaglini and B. Marinkovic. 'Synapse: A Scalable Protocol for Interconnecting Heterogeneous Overlay Networks'. In: *NETWORKING 2010 9th International IFIP TC 6 Networking Conference, Chennai, India, May 11-15, 2010. Proceedings*. Ed. by M. Crovella, L. M. Feeney, D. Rubenstein and S. V. Raghavan. Vol. 6091. Lecture Notes in Computer Science. Chennai, India: Springer Verlag, May 2010, pp. 67–82. DOI: 10.1007/978-3-642-12963-6_6. URL: https://hal.inria.fr/hal-00909544.

[9] F. Mallet and R. De Simone. 'Correctness issues on MARTE/CCSL constraints'. In: *Science of Computer Programming* 106 (Aug. 2015), pp. 78–92. DOI: 10.1016/j.scico.2015.03.001. URL: https://hal.inria.fr/hal-01257978.

[10] J.-V. Millo and R. De Simone. 'Periodic scheduling of marked graphs using balanced binary words'. In: *Theoretical Computer Science* 458.2 (Nov. 2012), pp. 113–130. DOI: 10.1016/j.tcs.2012.08.012. URL: https://hal.inria.fr/hal-00764076.

[11] G. Ngo Hoang, L. Liquori and H. Nguyen Chan. 'Backward-Compatible Cooperation of Heterogeneous P2P Systems'. In: *15th International Conference on Distributed Computing and Networking - ICDCN 2014, Coimbatore, India, January 4-7, 2014*. Vol. 8314. Lecture Notes in Computer Science. Coimbatore, India: Springer Verlag, Jan. 2014, pp. 287–301. DOI: 10.1007/978-3-642-45249-9_19. URL: https://hal.inria.fr/hal-00906798.

[12] D. Potop-Butucaru, R. De Simone and J.-P. Talpin. 'Synchronous hypothesis and polychronous languages'. In: *Embedded Systems Design and Verification*. Ed. by R. Zurawski. CRC Press, 2009, pp. 6-1-6–27. DOI: 10.1201/9781439807637.ch6. URL: https://hal.inria.fr/hal-00788473.

[13] M. Zhang, F. Dai and F. Mallet. 'Periodic scheduling for MARTE/CCSL: Theory and practice'. In: *Science of Computer Programming* 154 (Mar. 2018), pp. 42–60. DOI: 10.1016/j.scico.2017.08.015. URL: https://hal.inria.fr/hal-01670450.

10.2 Publications of the year

International journals

[14] F. Mallet and M. Zhang. 'Editorial - Theoretical Aspects of Software Engineering (2017)'. In: *Science of Computer Programming* 198 (Oct. 2020), p. 102521. DOI: 10.1016/j.scico.2020.102521. URL: https://hal.inria.fr/hal-03135431.

[15] D. Yue, V. Joloboff and F. Mallet. 'TRAP: trace runtime analysis of properties'. In: *Frontiers of Computer Science* 14.3 (June 2020), pp. 1–15. DOI: 10.1007/s11704-018-7217-7. URL: https://hal.inria.fr/hal-02402957.

[16] Y. Zhang, F. Mallet and Y. Chen. 'A verification framework for spatio-temporal consistency language with CCSL as a specification language'. In: *Frontiers of Computer Science* 14.1 (Feb. 2020), pp. 105–129. DOI: 10.1007/s11704-018-7054-8. URL: https://hal.inria.fr/hal-01924463.

[17] Y. Zhang, F. Mallet, H. Zhu, Y. Chen, B. Liu and Z. Liu. 'A clock-based dynamic logic for schedulability analysis of CCSL specifications'. In: *Science of Computer Programming* 202 (Feb. 2021), p. 102546. DOI: 10.1016/j.scico.2020.102546. URL: https://hal.inria.fr/hal-03135429.

[18] Y. Zhang, H. Wu, Y. Chen and F. Mallet. 'A clock-based dynamic logic for the verification of CCSL specifications in synchronous systems'. In: *Science of Computer Programming* 203 (Mar. 2021), p. 102591. DOI: 10.1016/j.scico.2020.102591. URL: https://hal.inria.fr/hal-03135428.

International peer-reviewed conferences

[19] P. Baufreton, V. Bregeon, K. Didier, G. Iooss, D. Potop-Butucaru and J. Souyris. 'Efficient fine-grain parallelism in shared memory for real-time avionics'. In: ERTS 2020 - 10th European Congress Embedded Real Time Systems. Toulouse, France, 29th Jan. 2020. URL: https://hal.inria.fr/hal-02431187.

[20] X. Chen, F. Mallet and X. Liu. 'Formally Verifying Sequence Diagrams for Safety Critical Systems'. In: International Symposium on Theoretical Aspects of Software Engineering. Hangzhou, China, 11th Dec. 2020. URL: https://hal.inria.fr/hal-03121933.

[21] P. Cuenot, P. Bouche, R. De Simone, J. Deantoni and A. Oueslati. 'Early validation of satellite COTS-on-board computing systems'. In: ERTS 2020 - 10th European Congress on Embedded Real-Time Software and Systems. Toulouse, France: http://www.erts2020.org/, 29th Jan. 2020. URL: https://hal.inria.fr/hal-02413867.

[22] F. Gao, F. Mallet, M. Zhang and M. Chen. 'Modeling and Verifying Uncertainty-Aware Timing Behaviors using Parametric Logical Time Constraint'. In: DATE 2020 - Design, Automation and Test in Europe Conference. Grenoble, France, 9th Mar. 2020. URL: https://hal.archives-ouvertes.fr/hal-02429533.

[23] Z. Hou and E. Madelaine. 'Symbolic Bisimulation for Open and Parameterized Systems'. In: PEPM 2020 - ACM SIGPLAN Workshop on Partial Evaluation and Program Manipulation. New-Orleans, United States, 19th Jan. 2020. DOI: 10.1145/3372884.3373161. URL: https://hal.inria.fr/hal-02406098.

[24] G. Liboni and J. Deantoni. 'A Semantic-Aware, Accurate and Efficient API for (Co-)Simulation of CPS'. In: *Software Engineering and Formal Methods. SEFM 2020 Collocated WorkshopsPrint ISBN: 978-3-030-67219-5Electronic ISBN: 978-3-030-67220-1*. CoSim-CPS 2020 - Software Engineering and Formal Methods. SEFM 2020 Collocated Workshops. Amsterdam / Online, Netherlands, 14th Sept. 2020. DOI: 10.1007/978-3-030-67220-1_21. URL: https://hal.inria.fr/hal-03038527.

[25] G. Liboni and J. Deantoni. 'CoSim20: An Integrated Development Environment for Accurate and Efficient Distributed Co-Simulations'. In: ICISE 2020 - 5th International Conference on Information Systems Engineering. Manchester / Virtual, United Kingdom, 20th Nov. 2020. URL: https://hal.inria.fr/hal-03038547.

[26] Q. Liu, R. De Simone, X. Chen and J. Liu. 'Multiform Logical Time & Space for Mobile Cyber-Physical System with Automated Driving Assistance System'. In: APSEC 2020 - Asia-Pacific Software Engineering Conference. Singapour, Singapore, 1st Dec. 2020. URL: https://hal.inria.fr/hal-02952919.

[27] Q. Liu, R. De Simone, X. Chen and J. Liu. 'Multiform Logical Time & Space for Specification of Automated Driving Assistance Systems: Work-in-Progress'. In: EMSOFT 2020 - International Conference on Embedded Software. Hamburg / Virtual, Germany, 20th Sept. 2020. URL: https://hal.inria.fr/hal-02952912.

[28] C. Stolze and L. Liquori. 'A Type Checker for a Logical Framework with Union and Intersection Types'. In: FSCD 2020 - 5th International Conference on Formal Structures for Computation and Deduction. Paris, France, 2020. DOI: 10.4230/LIPIcs.FSCD.2020. URL: https://hal.archives-ouvertes.fr/hal-02573605.

Scientific book chapters

[29] H. Zhao, L. Apvrille and F. Mallet. 'A Model-Based Combination Language for Scheduling Verification'. In: *Model-Driven Engineering and Software Development*. 2020. URL: https://hal.telecom-paris.fr/hal-02430903.

Reports & preprints

[30] R. Ameur-Boulifa, L. Henrio and E. Madelaine. *Compositional equivalences based on open pNets*. 8th Jan. 2021. URL: https://hal.inria.fr/hal-03103607.

[31] A. Ciaffaglione, P. D. Gianantonio, F. Honsell and L. Liquori. *A prototype-based approach to object reclassification*. Inria & Université Cote d'Azur, CNRS, I3S, Sophia Antipolis, France, 1st Dec. 2020. URL: https://hal.inria.fr/hal-01646168.

[32] S. Ghilezan, S. Kašterović, L. Liquori, B. Marinković, Z. Ognjanović and T. Stefanović. *Federating Digital Contact Tracing using Structured Overlay Networks*. 6th Feb. 2021. URL: https://hal.inria.fr/hal-03127890.

[33] G. Iooss, M. Pouzet, A. Cohen, D. Potop-Butucaru, J. Souyris, V. Bregeon and P. Baufreton. *1-Synchronous Programming of Large Scale, Multi-Periodic Real-Time Applications with Functional Degrees of Freedom*. 12th Mar. 2020. URL: https://hal.inria.fr/hal-02495471.

[34] L. Liquori, S. Wood and E. Scarrone. *Asynchronous Contact Tracing*. 14th Dec. 2020. URL: https://hal.inria.fr/hal-02989404.

[35] H. Pompougnac, U. Beaugnon, A. Cohen and D. Potop-Butucaru. *From SSA to Synchronous Concurrency and Back*. INRIA Sophia Antipolis - Méditerranée (France), Dec. 2020, p. 23. URL: https://hal.inria.fr/hal-03043623.

[36] B. Wang, E. Madelaine and M. Zhang. *Symbolic Weak Equivalences: Extension, Algorithms, and Minimization - Extended version*. Inria & Université Cote d'Azur, CNRS, I3S, Sophia Antipolis, France; East China Normal University (Shanghai), 30th Jan. 2021, p. 71. URL: https://hal.inria.fr/hal-03126313.

Other scientific publications

[37] L. Liquori, M.-A. Peraldi-Frati, A. Cimmino, S. M. Jeong, J. Koss and R. García-Castro. *SmartM2M; Study for oneM2M Discovery and Query use cases and requirements*. 2nd July 2020. URL: https://hal.inria.fr/hal-03115482.

[38] L. Liquori, M.-A. Peraldi-Frati, A. Q. Khan, A. Cimmino, S. M. Jeong, J. Koss, R. García-Castro, S. Kumar and S. El Khatab. *SmartM2M; Study for oneM2M; Discovery and Query solutions analysis & selection*. 1st Nov. 2020. URL: https://hal.inria.fr/hal-03115497.

[39] L. Liquori, E. Scarrone, S. Wood, F. Bob and L. Cees. *SmartM2M; Asynchronous Contact Tracing System: Fighting pandemic disease with Internet of Things*. 18th Dec. 2020. URL: https://hal.inria.fr/hal-02989793.

10.3 Cited publications

[40] G. Liboni, J. Deantoni, A. Portaluri, D. Quaglia and R. de Simone. 'Beyond Time-Triggered Co-simulation of Cyber-Physical Systems for Performance and Accuracy Improvements'. In: *10th Workshop on Rapid Simulation and Performance Evaluation: Methods and Tools*. Manchester, United Kingdom, Jan. 2018. URL: https://hal.inria.fr/hal-01675396.