Kairos

Joint Project-Team Proposal
(long version V8)

Université Côte d'Azur / Inria Sophia Antipolis - Méditerranée / I3S Laboratory (UMR CNRS 7271)

Inria Field: Algorithmics, Programming, Software and Architecture
Theme: Real-Time Embedded systems

Members

Julien Deantoni (MCF UNS) **jd**,
Robert de Simone (DR Inria, scientific leader) **rs**,
Luigi Liquori (DR Inria) **ll**,
Eric Madelaine (CR Inria) **em**,
Frédéric Mallet (PR UNS) **fm**,
Marie-Agnès Peraldi-Frati (MCF UNS) **map**,
Dumitru Potop-Butucaru (CR INRIA, Paris) **dp**,
Sid Touati (PR UNS) **st.**

Abstract

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) [1,2]. 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 "natural" *environments* and *execution platform* representations, which may become rather heterogeneous in the age of Internet of Things, with many-connected objects and computing *at the edges of the gateways*. The global resulting methodology can be tagged as Model-Based, Platform-Based, CPS engineering Design.

We shall describe below the research items corresponding to different design phases in more details, with the relevant investment of team members. We discuss the originality of our proposed approach, based on our previous expertise in real-time embedded system design. We comment on potential and existing collaborations, local or international, both with academic and industrial partners. We set up milestones for success, and assert risks within a reasonable horizon framework.

Cyber-Physical Systems have come to greater attention due to the growth of the digital part of society, with a profusion of connected objects, intelligent sensors, and microcontrollers of digital nature closely interacting with the physical reality [1,2,3]. Embedded systems have existed for a number of years, in car engine control, airplane flight management, factory automation, but the dense dissemination of sensors/actuators, and the coupling of such embedded control with communication capacities (as in the advent of smartphones and gateway computing), have drastically

enhanced the heterogeneous complexity of CPS design [4]. While various digital-only entities may communicate among themselves using dedicated protocols and synchronization standards, proper exchanges between *cyber* objects and their physical environment need to rely more on proper timing (*nature can't wait*). Modeling then must find the proper trade-off between accuracy and efficiency (of simulation, or execution…). In addition, the nature of the underlying computation and communication infrastructure may have a huge impact on *cyber* subsystem realization. For instance, *always-on* connected objects may be subject to drastic limitations in power consumption.
From this, it turns out that CPS design must take into account all three aspects of application requirements, execution platform guarantees and contextual physical environment to establish both functional and temporal correctness.

> **Main challenges for Kairos.** The Kairos proposal ambitions to deal with correct design of Cyber-Physical embedded Systems as a whole. This requires joint modeling of distinct and heterogeneous aspects: 1) cyber platforms, together with 2) their physical environments, but also 3) abstract modeling of applicative software. Indeed, a main concern in Kairos is to couple tightly these models, in early specification stages, so as to predict as accurately as possible system-level properties of applications mapped onto platforms: functional and temporal correctness primarily, but also a range of Non-Functional Properties (NFP) possibly. While these types of modeling are frequent engineering practices in the physical domains, the (cyber) program abstraction is less commonplace in software engineering. The kind of embedded applications we target are generally coined as "reactive" programs, for their interactions with physical environments, and they lead to program abstractions as communicating processes and executable specifications.
> This overall goal is referred to as Model-Based System Engineering (MBSE) [27] or Platform-Based Design (PBD) [26] in distinct research communities. Currently there exist in this area both efficient point-wise tools and methods on one hand, global methodology frameworks on the other hand. All of them rely on some forms of modeling, and this profusion of models is far from providing a consistent scheme altogether: point-wise analysis tools will often rely on specific models, involved in domain-specific languages (DSL), with unclear or ambiguous dynamic semantic interpretation; methodology frameworks will also often provide incomplete models, to target early design stages more freely. Our goal will be nevertheless to benefit from all the knowledge and industrial-level experience embodied in these models and environments (and certainly not to start from scratch again). This exposes the need to complete and harmonize these models and their related domain-specific language representation, so as to reach operational semantic seamlessness.

We address these challenges using the concept of **"logical time"** as a central notion [20,5,6], to provide constraints between modeling elements that remain flexible, to be fixed only later in the course of design decisions (after allocation mappings). Rather than being stand-alone, these constraints may be used to complement models on their dynamic operational aspects, completing information that may be lacking originally. Logical time can also be used as an intermediate between non-functional property constraint formulation, and their physical time solution. Reuse in design is also promoted by usage of logical time formulation. At this point, it may be useful to provide an introduction and motivation for the notion of Logical Multiform Time (and Logical Clocks [5,6]) which is central in Kairos methodological approach, as it can lead to misconceptions if not properly introduced.

Logical Multiform Time. The idea of Logical Multiform Time in System Design is based initially on the

simple remark, that conceptual design is in many case based on the recognition of those meaningful timed events whose occurrence guide the future system evolutions. The one important thing to note is that actual physical timing may generally not be known at design time, or even that it may later differ greatly in various run-time executions. Thus, design has to deal with variable time parameters in what we term "logical events". As such events may often occur repeatedly, we adopt the naming of Logical Clocks for sequences of event occurrences, although care as to be taken not to confuse such Clocks with Watches measuring physical times. While logical time can (also) be regularly linked to physical time (periodic), as often in sensor samplings, fixed-step time-driven simulation, or simple processor (quartz) clock cycles, it is not necessarily so at design stage: fancy processors may change their speed, 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…). See lectures by Gerard Berry at College de France (https://www.college-de-france.fr/site/gerard-berry/course-2013-2014.htm) or seminal work by E. Lee and A. Sangiovanni-Vincentelli [14] for a motivation of multiform logical time in embedded and CPS design.

In the Logical Multiform Time approach, specification of designs will then essentially consist in providing: -requirements, to hold between Logical Events and Clocks (to be guaranteed of systems); or conversely, - provisions (to be assumed of systems) describing established logical time relations. Such constraints are only partial, and may allow degrees of freedom, to be resolved later as potential final physical mapping. Now, what gives our Logical Time approach its specificity is that we do not only provide a time model for semantics definition outside the specifications themselves, but instead we provide a consistent language of operators and constructors on Logical Events and Clocks, combining their effects with a large (dedicated) expressivity. As a restriction, we stick so far to discrete, "deterministic" time as opposed to stochastic time models.

This resulted in the Clock Constraint Specification Language (CCSL) [AM08,And09]. In CCSL one may for instance state that a clock is a subclock of another one, according to a given occurrence pattern, or that a clock is faster than another one, to give just a glimpse of the language expressivity. Rooted in these two partial orders, clock constraint primitives of the language allow to express most of the practical time models encountered for Models of Computation and Communication (MoCCs) relevant to our domain.

Maybe this deserves to be underlined strongly: CCSL does not define one time model, but ambitions to construct time models for particular applications according to the designer's plans. The MoCCs considered in our works for supporting the logical time annotations for precise semantic constraints have so far consisted of task graphs and data-flow process networks (action-based models), coupled with hierarchical state-based components for modes. While the choice of CCSL primitives was historically indebted to needs found in Synchronous Reactive Languages [PST09,PSST11], Classical (Process Network)[MDAS10] and Real-Time Scheduling [MiSi12], care was taken to make it agnostic: the intention was indeed to provide a second-order definition formalism where to express the actual timed dynamics for existing or new MoCCs under our consideration . Examples range in various specification formalisms such as Synchronous Reactive Languages [28], Data-Flow Process Networks (SDF and beyond [4), Kahn and Petri Nets, task models for Classical and Real-Time Scheduling Theories (periodicity), AADL and AutoSar for industrial sources... Of course, the utterly important task of solving those constraints expressed in CCSL in each case is another business than merely representing them (which takes design skills).

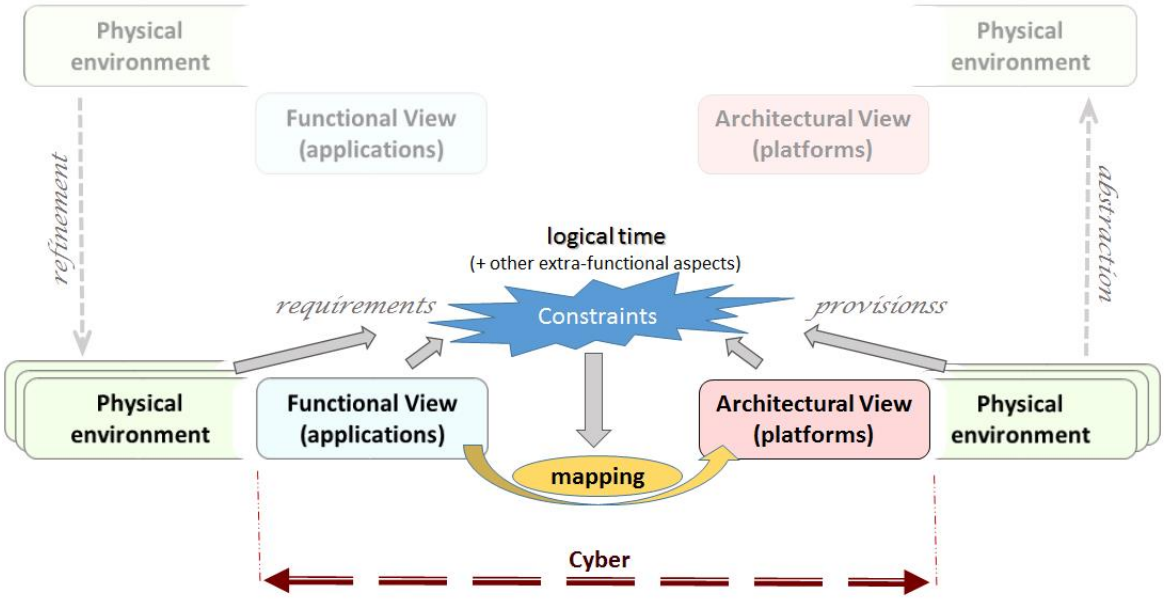
The main practical application of Logical Multiform Time was in Platform-Based design, where joint modeling of applicative functionality and architectural platform views are provided, with mostly

applicative requirements and mostly architectural provisions. Solving the combined constraints amounts to perform an abstract model-level mapping (scheduling and allocation); this approach was formerly promoted in Aoste as Application/Architecture Adaptation (AAA) methodology http://www.syndex.org/, and this will be extended in Kairos (see also [Potop15]). Note again that exact physical time need not be known precisely at that stage to allow mapping based on Logical Multiple Time constraints.

So far, one can recognize the benefit of Multiform Logical Time in its ability to express precise occurrence constraints describing essential system features, without imposing actual physical timing constants. As such, it should not be neglected, if only in formal specification purposes. Still, we ambition of course to be able to apply efficient analysis, verification and optimization techniques whenever possible. Constraints (on infinite sequences) comprise combinations of Boolean logic and integer arithmetic, so that SMT-solvers with relevant theories may be considered, either by recognizing existing timed models in which such techniques are already available, or by producing ourselves new such techniques on precisely defined syntactic subclasses.

CPS systems calls for new challenges due to their "hybrid" nature, indicating how discrete/cyber and continuous/physical processes coexist in design. New induced phenomena (uncertainty, mobility) may call for reflection in added CCSL primitives.

Our general strategy is to promote the insertion of the logical time dimension, with its openings towards realistic formal methods and related verification techniques, into some of the mainstream Model-Based Embedded System Engineering (MBSE) methodologies, such as SysML/MARTE, ARCADIA/Capella, AADL. While this objective was already central in the former Aoste project-team that preceded Kairos, the new emphasis on Cyber-Physical Systems, as extensions of Real-Time Embedded systems, extends greatly the scope of platforms and models considered.



The figure above describes the role ascribed to logical time notions as support for linking the dynamics of the various CPS ingredients. While logical time is firstly meant to decorate functional and architectural elements of applications and execution platforms altogether, it can serve as useful support for non-functional requirements and provisions (for instance, power consumption,

temperature, or performance, are accumulated values over activity periods expressed by designers using “virtual” logical time).

The current agenda of Kairos is twofold: first, to deepen and consolidate the impact of our approach by real-life experimentations that always provide useful feedback on new practical and theoretical topics; second, to extend our work with new concerns in CPS systems, and wider scope in Internet of Things and Smart-* platform models. These add several demands to those addressed so far: 1) increased heterogeneity in models and device components, and in DSL composition; 2) uncertainty of temporal conditions in probabilistic and stochastic cases; 3) dynamicity and mobility in platforms. We are effectively working on specific application domains with prominent industrial partners (Thales, Safran, Thales Alenia Space, Airbus), and we plan to extend our scope in the fields of Intelligent Transportation Systems (in collaboration with Renault research lab in Sophia) and Languages for Smart Contracts (in collaboration with Accenture Labs in Sophia). The growing demands for strong correctness in certification processes for those safety-critical domains will further trigger the need for sound mathematical design methodologies as projected in Kairos.

Our activities are supported by practical prototype tools to conduct (meta) modeling, analysis, verification and/or synthesis. Our concern in compliance with mainstream MBSE environments is reflected in the fact that these tools can mostly be used as plug-in extensions in industrial-strength frameworks, but also as stand-alone techniques. We shall indicate during the description of our research directions, which are to be reflected in tool building (mostly expanding our current toolset). A more systematic description of tool features will be provided later.

The document is organized as follows: first section describes our planned research directions; section 2 provides a view of our current collaborations, both academic and industrial; section 3 provides a self-assessment of goals and measures of success in short to medium term, taking into account state-of-the-art practices; last sections recall existing tools developments and team member curriculum before a selected bibliography.

1 RESEARCH DIRECTIONS

We shall describe next how we plan to articulate our efforts within Kairos to achieve the previous objectives. For convenience, we split our presentation in several subsections:

- a subsection on co-modeling deals with extensions and adaptations of formalisms previously developed by us and our neighboring communities, insisting on logical time background and CPS target;
- then another subsection on general Model-Based System Engineering dedicated to CPS and rooted in formal methods;
- then a subsection describing the analysis, verification and validation (V&V) formal methods studied on those (extended) models; these are meant to comprise useful scheduling, mapping, and co-simulation activities for instance;
- finally, a last subsection describing the potential relations between the former model/specification level dealt with in previous parts, and real implementations; those relations may go both ways: first of course by studying the feasibility of code synthesis from co-simulation and allocation results, whenever feasible; second, by computing information on concrete individual components to be lifted as parameter values back to modeling level.

Of course, there is a strong linkage between the different parts, modeling being adapted purposefully as to allow and promote the algorithmic methods as part of a global design methodology. Still, hopefully the proposed presentation favors readability by introducing these in a stepwise fashion.

1.1 Logical Time and CPS Modeling extensions fm, rs, jd, map, st, dp, ll

Note: the various work items mentioned below will aim at extending the existing CCSL formalism with adequate (and hopefully natural) constructs for the larger CPS and IoT specification scopes. While extensions may be diverse, they will always try to apply later on mainstream modeling framework, to provide precise formal interpretation on top of existing methodological diagrams.

Logical time extensions (for uncertainty/variability). Considering Cyber-Physical Systems as a network of embedded systems interacting with their physical environments introduce a fundamental shift in the design method [DHJM18,LLHM13]. While reactive embedded cyber controllers strive to be predictable and accurate, the environment is generally less predictable, with multiple sources of timing uncertainty. This is especially reflected in practice at the level of multiple sensors and actuators, where the time rates of physical and cyber subsystems are connected.

In addition to environmental inputs, the computation platform itself may display physical time variability (cache misses, imprecise worst-case execution time...). Failure modes, so important in embedded and CPS computing and dysfunctional Safety Analysis are other important sources of stochastic/probabilistic time modeling imperatives.

> The main challenge here is the definition of relevant dedicated stochastic/ probabilistic combinators and constraints between logical clocks.

Logical time extensions to support Non-Functional Property. A primordial aspect of Platform-Based Design is that, while models of applications and architectural platforms are provided independently, they are to be adapted to later to one another, seeking to optimize or respect some criteria. Functional correctness of course, but Non Functional Properties such as performance, power consumption, affordable temperature, levels of fault tolerance (in Safety Analysis) or security (in privacy/affinity constraints) are typical examples of criteria to be taken into account, in requirements requested by the applicative parts or guarantees offered by the CPS platforms [GoDM13,KhRM17]. These NFPs are often mutually correlated and sometimes even antagonistic (performance vs power consumption vs temperature for instance). We feel that the notion of logical time may play a privileged role here for expressing relations that may be time-dependent, but in ways that indeed abstract from the pure physical time (again, for instance, faults may occur relative to the NFP conditions of a components being used in the past).

> As in the previous case, the challenge here is to provide combinators for logical events, understood as providing indirect information on non-functional quantities, abstracted from physical time.

Extensions for spatio-temporal modeling and mobile systems. The previous concern for modeling of platform infrastructures induces a concern also for spatio-temporal aspects. 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 CPS (or mCPS) systems occur casually, 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 expressivity will certainly cause new needs in handling constraint systems and topological graph locations.

In Kairos, we shall study extensions to multiform logical time formalisms able to encompass spatio-temporal specifications. The goal here will largely be to abstract topological aspects originated from mobile platforms and applications specifications, into an integrated set of constraints that can lead to dedicated analysis and optimization techniques.

Note that extension of rigourous design methods to the era of dynamic and mobile IoT CPS has recently been strongly advocated by Joseph Sifakis in [21].

> The main challenge here 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 as described in the next section.

1.2 Formal CPS design *jd, em, fm, ll, rs, map, dp*

Note: while the previous section focused on extensions to logical time notions, this one exploits their intention to deal with a wider scope of systems, targeting IoT CPS domains.

CPS System Engineering as a design methodology is generally established on a set of widely accepted diagrammatic representations, such as (functional or architectural) block diagrams, hierarchical state and activity diagrams, control- and data-flow graphs etc. Although generally popular as graphical syntax, they usually come also with textual counterparts, but rather low-level. Consider Matlab/Simulink, SysML/MARTE [MPDS14,Mall15], AADL, or Capella [BCEL15] as instances. As an immediate problem, the operational semantics of these diagrams (the way they behave dynamically) is either incomplete and vague on generic environments, or ad-hoc and obscurely specialized often on specific pointwise tools and methods (which themselves may be part of a larger, generic framework). This reflects a discontinuity between the activities of global system architects, and specific component designers.

Our long-term approach aimed at promoting diagrams to the status of Domain-Specific Modeling Languages (DSMLs), by providing means to express the semantic temporal rules governing their dynamics. An initial goal is to be able to cover prominent Models of Computation and Communication (MoCCs) as found in the literature (on Concurrency Theory mostly). A further goal is to be able to define the composition and the behavior orchestration of such heterogeneous MoCCs co-operating, even when mixing discrete and continuous cyber/physical models..

The role of architectural platform and non-functional property modeling should not be forgotten here, as many functional applications will in fact be represented first as a platform-independent model, with some (restricted) amount of freedom regarding time captured in logical time requirements, while decisions on scheduling and mapping of that functionality onto a specific execution platform infrastructure remain delayed. Indeed, this is the primary motivation for the use of logical time, that it allows the system designer to free her/himself from the burden of exact timing (that may evolve and change according to system versions), while keeping a precise way to maintain formal compliance with the final adequation.

Our efforts on Formal CPS design will be backed by tools, with one side aimed at providing the proper representations described here, and another side on associated methods included in the next section (we split because in some cases alternative methods may be used on the same descriptive models).

Former research directions previously conducted in the Aoste EPC on Platform-Based Design and

heterogeneous DSMLs will be continued and expanded to the CPS and IoT domains. Thanks to the addition of new members we will extend towards the study of the equivalence of semantic definitions (open term bisimulations), towards integration of logical time properties in object-oriented languages and logical frameworks, and towards mobile CPS.

Heterogeneous Domain-Specific Modeling Languages (DSMLs). The focus here is on the coordination, and properly defined behavioral combination, of existing or new Models of Computation and Communication incorporated in DSMLs. The CPS dimension adds to the problem of coupling MoCCs that are strongly heterogeneous in nature. We plan to extend our GeMoC studio environment according to this larger scope [Dean16, DDTC15,VLDC15].
Work here is strongly inspired from the Ptolemy environment from UC Berkeley [17]. But, while Ptolemy only relies indirectly on the formal semantics (execution engines are reimplemented manually to satisfy supposedly the semantic definition), GeMoC runs exactly the semantic constraints that are the true MoCC/DSML behavioral definition.
An important issue here is that technological DSMLs and theoretical (formal) MoCCs are developed and considered initially in distinct communities, so that part of our activity consist in reconciling the two worlds. This becomes even more so with the introduction of physical models and complex co-simulation.

> Our main challenge here is to confront our approach to a large number of practices in system modeling, trying to establish the benefit of expressing explicit (logical, variable) precise timing relations for the semantic definition of DSMLs.

A distinct topic we plan to investigate in this domain is the relation between (formal concurrent) MoCCS and parallel programming languages (not to be confused with DSMLs) such as MPI and OpenMP, which will certainly provide instances of applicative functionalities. Other languages involved in reactive programming may also be studied, but this remains speculative research depending on our labor ability.

Platform-Based Design (PBD). Most of the needs for representation of architectural platforms and NFP requirements/provisions information could be supported by existing MBSE environments with their mechanisms for completion (SysML profiles and stereotypes, Capella views...), and we shall maintain some level of expertise in this. In a more focused area, where we want to face the issue of code synthesis (see next section), we shall continue the efforts of the LoPHT tool, again with extension towards the CPS general area. As already stated, the prominent role of sensors/actuators, in this case where interactions with physics may be distinct than pure cyber communications, makes the approach highly sensible to the material aspects of cyber interactions with physics.

> Our challenge here is to greatly enlarge our scope of platform modeling, extending it to CPS IoT domain, in which devices mediating the cyber and physical parts may be of uttermost importance [26]. A further point of consideration may be that, while Logical Time may be used at design time to initiate platform-based mapping concerns, it can also be regarded at later stages of system operation to record information that has been learned at runtime inside the updated models themselves. This aspect is increasingly considered inside the notion of “digital twins” (or sufficiently complete models to span successive product life developments. The notion of learning by actual use in real situations, while utterly simple and less spoken of than specific deep- learning techniques in current AI , has certainly deep roots both in adaptive control theory and in planning altogether.

SoS semantics and open systems. An important conceptual tool for the precise and sound definition of language semantics, especially in the area of Concurrency, is that of Plotkin's Structural Operational Semantics (SOS) rewrite rules. We have started developing a general approach (using a low-level formalism named pNets), with the long term goal to provide both a semantic framework, and a verification platform, allowing for the analysis of CPS systems, and more generally complex systems featuring communication, synchronisation, and explicit handling of data, time, locations, etc. [MaZh16,HeMZ16]. The very essence of logical time constraints, which does not force exact timing but allows multiple schedule solutions in general, will certainly be a source for behavioral equivalence. Relevant analysis techniques will be described in next section, and implemented in our Vercors tool.

> Our challenge here is to study generalized bisimulation notions that characterize behaviors where distinct data abstractions and/or logical timings amounts to equivalent functional effects.

Object-oriented programming and logical time. As already mentioned several time, we view our logical time model as a mean to enhance the description of timing constraints and properties on top of existing specification formalisms. When considering general-purpose object-oriented languages like Java, type-theory is a natural way to provide such properties. Currently, such languages do not have constructs nor special types to manage instants, time structures and instant relations like subclocking, precedence, causality, equality, coincidence, exclusion, independence. The Clock Constraint Specification Language provides such *ad hoc* constructors to specify clock constraints and logical time, so that enriching object oriented type theories with CCSL expressions constitute an interesting research perspective towards a wider usage of CCSL. Zelus [15] and SystemJ [28] was suggested to us by reviewing experts as a step in a similar direction.

> Our challenge here is to study the use of logical time constraints as behavioral type properties, and the design of programming language constructs and ad-hoc type system.

1.3 Analysis, Verification & Validation **jd, em, fm, ll, map, rs, dp**

The two previous sections allow us to describe and extend to the CPS case the relevant notions of logical time and related notions, applied to CPS and IoT platforms and MoCCs for applicative functionalities, in the goal of describing in more formal and precise terms the temporal (and other non-functional) dimensions of the systems. Now we want to apply analysis and verification methods, first to be able to co-simulate and animate timed specification, then to establish correctness properties by exhaustive construction of traces and reachable configurations (if feasible), last to propose optimized adequation mapping between applicative functions and architecture platform; this can be fully automatic, or using user guidance.

Co-simulation. From the previous constructions, a complete CPS model comprises a number of distinct MoCCs embedded in DSMLs, specifying altogether applicative functionalities and architectural platform features, with physical environments, all of them endowed with constraints and provisions expressed by the means of logical time properties. Individual MoCCs may be continuous- or discrete-time, leading naturally to simulation style that are time-triggered or event-based respectively. Coordination of individual MoCCs by orchestration, building heterogeneous hierarchical systems, is also expressed using logical time constraints [VLDC15]. Now the challenge is to devise a correct and efficient co-simulation framework combining all of these.

The FMI/FMU framework builds what is called there a *Master Algorithm* to realize the needed composition of simulators corresponding to individual MoCCS, engineered to respect semantic

constraints. Our case is made more complex because we allow both continuous-time (physical) models but also cyber discrete models, so that combinations of time-triggered and event-driven simulator progress is requested; in addition the potential composition due to logical time constraints may be richer [CeDS16,LDPQ18]. This requires more refined techniques to build the new Master Algorithms, and then use them to obtain timed traces as the applications of heterogeneous co-simulation. We will further work in this direction.

The further extensions to probabilistic/stochastic time models, mobility and spatio-temporal aspects will be progressively addressed in the course of the project. In addition, the specific role of platform provisions (as opposed to requirements) in constraints will be studied to improve simulation performance. Corresponding extensions of the TimeSquare tool should be used to support these developments, when attempting to generate traces from the constraint sets and the Master Algorithms.

> The main challenge here will be to describe formally and univocally as a Master Algorithm the efficient coordination coupling of individual simulation models that may comprise both time-triggered and event-based classes of models.

Optimized CPS Platform mapping. While the previous co-simulation scheme tends to put all executable models (of applicative functionalities, physical environments and architectural platforms altogether) on an equal footing, the mapping allocation made popular in Platform-Based Design and the AAA (Adequation Algorithm Architecture) methodology claims to study the *directed* mapping of the functional application onto the cyber (sub)platform, in the context of physical environments. Such mapping will not in general depend on the dynamic simulation of the behavior to operate "at run-time", but rather work on a static version of the application, possibly finitely unfolded [KoSi16,MiKD15].
In practice, the application model is restricted to a homogeneous MoCC nature, in order for the mapping to remain tractable despite the range of potential choices. Actual mappings comprise both scheduling and allocation decisions, which are typically solved by optimization according to the extensions planned in scope. A simple starting point would be to consider physical environments that describe sensors/actuators periodic sampling rates,and then further patterns of arrival rates could be provided as CCSL requirements..
The LoPHt tool will support some of these developments. It is currently connected to the SCADE (synchronous) formalism as input MoCC, and should be better linked to MBSE platform modeling in the future.

> The main challenge here is to apply the results obtained in the previous research items in the larger AAA framework required by the CPS IoT context.

Model-checking and automatic solvers. Many of our analyses will consider solving sets of constraints of specific shapes, in particular to exhibit the existence of feasible runs (schedules) satisfying the logical clocking conditions. Extended simulation schemes, as described above, can verify that a human-provided solution is indeed valid. However, in many cases (when the domain of reachable configurations is regular enough), one can try to do better, by exploring more exhaustively the set of all possible runs. This is amenable either to model-checking or to constraints solving in dedicated axiomatic theories. Recently the two approaches have been used in combination with so-called SMT solvers (Satisfiability modulo theory). Here we will mainly try to study how to extend or combine existing formalisms taking into account the specifics of our genuine logical time notions [ZhDM18,MaSi15,MPDS14,MaMD13], resulting in hybrid approaches. An important feature here is

that constraints in our extended CPS case may be divided between required and guaranteed ones, so that the later can be used to prune the search while reflecting assumptions on the target platform to the model level. The combination with operation research techniques [ZhDM18,KoSi16,GKCP15,YLDM13,MiSi12] may also be extended considering the nature of external models..
While we do not consider any form of temporal logics based on logical time, the mere language for logical time constraints in itself allows expressing additional constraints (possibly involving states and modes) that will narrow the reachable state space, hopefully drastically. A typical instance of this arises when the difference of two logical clocks may grow unbounded and requires (increment/decrement) integer counters in modeling; this feature may cause Turing-complete expressiveness to the language; so additional restricting constraints, expressed as observers, may be requested as possible way to enforce boundedness, and thereforer model-checking decidability. More work will be continued in this direction.

Regarding probabilistic/stochastic verification models, various formalisms have been proposed to cover probabilistic model-checking in the case of traditional (dense) physical time. Timed Automata, then Hybrid Automata and their probabilistic and stochastic variants, have been used to define actual model-checkers in various academic places. Such models could be combined in several ways. For instance they could notify their meaningful logical events, such as zero-crossing, to be used as logical clocks by a surrounding environment entirely based on logical time constraints (such as for the Master Algorithms of co-simulation).

Our TimeSquare tool provides ways to generate exhaustive behaviors in favorable case for CCSL specifications, but needs to be extended according to intended extensions in Kairos. In many cases, this will most probably mean relying on auxiliary public-domain prover engines (as for SMT solvers).

> We shall study how to provide these verification functions from other tools implementing the relevant classes of model-checking techniques, targeting the additional expressiveness in models as described in previous section.

Behavioral semantics and bisimulation for open systems. Our current main challenge now is to understand how algorithmic pieces (computing and minimizing state-spaces, model-checking or equivalence-checking) can cooperate with theory specific reasoning (data, time, non-functional aspects) that will be delegated to some Automatic Theorem-Prover (ATP) or Satisfiability Modulo Theory (SMT) engines. Probabilistic bisimulation may also be a target for uncertain logical time in CPS.
The pNet model introduced in previous section has the proper structure to distinguish between control and data (and possibly cyber and physical) aspects, while keeping the control part described in a finite way, allowing for decidable algorithms for computing the behavioural semantics, for bisimulation equivalence checking, and for model-checking. Here the challenge is that the data part of the model (including time and non-functional aspects) must be handled in a symbolic fashion, and the reasoning engines must cooperate tightly with the (finite) algorithmic part.
We are starting developing these algorithms within the VerCors platform with (finite) symbolic automata representing open systems, using the Z3 SMT engine for satisfiability checking [QBMZ18].

> The challenge here is in the efficient algorithmic representation of data sets and time constraints to be held in the symbolic bisimulation checking.

It may occur to the reader that some of the goals of this work item are overlapping with the previous

one, verification being the common ground. They even share to some extent technical apparatus and resolution methods. Still, the more refined specificity of each concern, and the separate nature of past developments historically (Vercors vs TimeSquare) makes it likely that, while joint progress may be sought for inside developments, the two efforts will retain their external presentation in tool development, rather than merge in something still indistinct in scope.

Extending logical frameworks with logical time. The Curry-Howard isomorphism will be investigated to study logical time constraints within a dependent type theory. Dependent-types such as lock-types, developed in the past [HLMS17,HLMS18], can express the fact that in order to obtain a term of a given type S it is necessary to verify the *constraint P* (e.g., *if event A happened-before event B, then the timestamp of A is less than the timestamp of B*). We plan to extend the *Edinburgh Logical Framework* (LF) of Harper-Honsell-Plotkin to provide relevant constructs expressing logical time and synchronization between processes. Also, union and intersection types with their subtyping theories could capture some CCSL constraints needed to formalize logical clocks (in particular CCSL expressions like *subclock, clock union, intersection and concatenation*) and provide opportunities for an extended type theory [DLLS16,LiSt17,SLHS17,HLSS18].

We are convinced that logical time constraints seen as property types can be beneficially handled by the frameworks just described. Still, the topic remains more novel than others. We plan to use Type Theory and relations like subtyping to describe “tick” and relations between events. The slogan we have in mind is “tick-as-types”. As said above, we are also thinking to other type disciplines, using e.g. type-inheritance and dependent-types. Logical frameworks as the Edinburgh LF one could be improved with this kind of “modal” types. We could expect the formalization of logical time related properties simpler.

> The challenge here is to demonstrate the relevance of type theory to work on logical and multiform timing constraint resolution.

1.4 From models to implementations (and vice-versa) st, dp, rs

The computation of an abstract mapping (both scheduling and allocation), or a co-simulation environment are only a step towards real-implementation on a true CPS platform. While the functional application may be considered as abstracting software code, and as such could effectively contribute to producing the implementation version, the architectural platform and the physical environment models are only reflection of a pre-existing reality that must not be synthesized but checked for compliance with the model assumptions.

On the other hand, individual components from reality may be used to determine the actual values of logical time parameters, lifted back in the model design. A famous example of that is the so-called Worst-Case Execution Time (WCET), which provides an upper bound for the computation duration of a task or function. WCET may be obtained by over-approximation of the sum of smaller instructions, or of computation times measured in real situations.

Synthesis. To allow the automatic implementation of complex embedded systems, we advocate for a *real-time systems compilation* approach that combines aspects of both real-time scheduling and (classical) compilation. Like a classical compiler such as GCC, a real-time systems compiler should use fast and efficient scheduling and code generation heuristics, to ensure scalability. It should provide traceability support under the form of informative error messages enabling an incremental trial-and-error design style, much like that of classical application software. This is more difficult than in a classical compiler, given the complexity of the transformation flow (creation of tasks, allocation, scheduling, synthesis of communication and synchronization code, etc.), and requires a full formal

integration along the whole flow, including the crucial issue of correct hardware abstraction.

A real-time systems compiler should perform precise, conservative timing accounting along the whole scheduling and code generation flow, allowing it to produce safe and tight real-time guarantees. More generally, and unlike in classical compilers, the allocation and scheduling algorithms must take into account a variety of non-functional, usually safety-critical requirements, such as real-time constraints, preemption, mixed-criticality partitioning, allocation constraints and affinity, etc. Here the focus is put only on strict satisfaction of requirements, rather than optimization of a multidimensional metrics as in the case of classical compilation; resulting scheduling problems are differently stated.

The Lopht tool [Pot15,CP14,CPSL15,CDPS14] targets this new kind of real-time compilation. We will continue its development, while hopefully exploiting and integrating some of the modeling and analysis results of sections 1.1-1.3, and defining new associated synthesis methods. The use of automatic solvers [GKCP15] and constraint programming may offer more flexibility for applications of moderate size case where the mapping problem itself may evolve rapidly. The use of statistical methods to determine the best options of existing tools (e.g. the best combinations of compiler flags and parameters [ToDi14]) is another direction we will follow. This code generation phase should be seen as a separate system-level compilation phase independent from classical (e.g. gcc) compilation, and strongly rely on the work of the previous section to perform optimized mapping of the application on parallel and concurrent platforms. Of course, this questions the faithfulness in modeling, both in platforms and in models for abstract representation of programs. The CPS dimension also increases the demands on accurate handling of interactions with the physical world that must be validated.

Code performance analyses and predictability improvement. As noted above, running the implementation code of individual components may lead to establishing useful values, such as WCET (in logical time) that may be then used to characterize task and function durations. But we want to go beyond this worst-case (or average-case) timing analysis, and consider the issue of code variability (or execution time variance), which are a primary source of interest when dealing with real-time and safety-critical applications, rather than best-effort [WoTo16]. Low performance variability results in general from the good adaptation of the application to the architectural platform, this time at implementation level. There are still interesting issues:

- how to understand the precise factors that influence program performance variability, and how to model them;
- how to use this code performance model to make useful predictions about future executions. This can be done in two ways: by observing the code performance of past executions in order to build statistical models for future executions, or by optimizing the code for better mapping choices onto hardware/software platforms.

While this work considers primarily generic advanced optimizations for code implementations [ToDi14], it will seek usefulness in our general model-based approach, primarily by attempting to drastically reduce time variability in execution, so that WCET estimations fed back to model-based design stages for components are actually as accurate as possible.

The main purpose of this section is to feed models with information on elementary components that is as accurate as possible, to empower the model analysis of global systems

that fulfils the requirements of previous research items in return.

Summary and comments on research directions

We hope the previous research item descriptions displayed enough information to motivate their insertion in a general design approach based on Logical Multiform Time. They are organized in a loose methodological flow fashion, and require sometimes distinct expertises (see section 6). Some are to be involved from the previous Aoste project-team, from which many Kairos members are originated. In all cases, they imply extensions of the scope, and in some cases open new speculative directions. In this sense not all these work items should be viewed as equally important (in workload, not in interest), or as having the same predictive horizons for tangible results (see section 3). In fact, it should be mentioned that the preparation of Kairos has prompted an increase in joint technical background knowledge between new and former members, which spread also to the junior, non-permanent staff (PhDs mostly). Still, we view the diversity of competencies amongst team members, considered in the light of globally common goals, as a strength of Kairos.

2 COLLABORATIONS AND PARTNERSHIPS

2.1 Academic community

As already mentioned, our theoretical background in formal modeling comes mostly from the semantic of programming languages (especially synchronous and object-oriented), process networks, real-time scheduling, model-based system engineering, type theory and advanced compilation. We are still strongly active in the corresponding international communities (maybe less in real-time scheduling where the main expertise stayed with former Aoste-Paris members) and the related conferences and special events.

Inside Inria, we have strong ties with other groups inside the specific theme on real-time embedded systems (EPIs Parkas, TEA, Spades, and HyCOMES), but also in model-driven engineering (EPI DiVerSE), and others more informally (such as CASH and SPIRALS). The premises of the Logical Multiform Time approach are shared with TEA (Signal/Polychrony), Parkas (Lucid Synchrone, N-synchrony), Spades (Process Networks, Timed Models), HyCOMES (Contracts), and we share in part a common theoretical heritage.
Moreover, the Zélus hybrid simulation tool developed in Parkas, or liquid clocks as studied in Tea, or even Precision-Timed Machines from Spades, are all heading in the same general direction for the future of associating logical and physical time for CPS modeling. Our originality in this community is to propose an independent language for time-model constructors, to be embarked in model-based system engineering methodologies: we aim at domain-specific modeling perhaps firstly, even before than model-specific solving of constraints in a given individual (promising) case of formalism.
We share partnership with HyCOMES and DiVerSE in the Glose industrial collaboration, with Parkas in the ASSUME ITEA3 project, and in the past we shared a LIAMA collaboration with TEA (see contract section for the description of these collaborations). We have applied alongside these teams to other funding schemes as well. The yearly open workshop Synchron, which we organize in turn together with mostly german partners is a renewed opportunity for Inria teams of the Real-Time Embedded theme to exchange on technical topics for a full week seminar.

We are starting participation in an IPL (Inria Project-Lab) named SPAI (Security by Program Analysis in the IoT), headed by the Indes EPI. There our role concerns the model-based design of IoT environments, while other Inria partners (EPIs Antique, Celtique, and Privatics) deal with formal analysis and extensions of JavaScript descriptions for programs evolving in this context, focusing on security aspects.

Importantly, Kairos will remain part of the ComRED team of UMR I3S, which includes Coati and former Scale Inria teams as well (note that CNRS lab team structures are thus strictly larger than Inria's). Furthermore, we maintain at the local level (in the Nice/Sophia Antipolis area and around the Université Côte d'Azur perimeter) multiple interactions with other teams of UMR I3S and UMR LEAT (on smart contracts), as well as Telecom ParisTech LabSoC team (on security apsects in Model-Based System Engineering). This led to joint partnerships in larger initiatives (see below), but also to joint initiatives towards the more local Labex UCN@sophia (joint PhDs with LEAT and Telecom ParisTech for instance). Local efforts shall be continued in the upcoming EUR ("Ecole Universitaire de Recherche"), part of the emerging strategic axis named *Digital Systems for Humans* carried by the RISE Academy of UCA Jedi. This close partnership should mix research with education programs, at Master level. We view it as important since our research program requires a deep integration of people with complementary skills, from antennas to system engineering, including the underlying execution platforms and the ad-hoc short to wide range communication protocols (e.g. Lora).

Another active collaboration (inside ComRED with Scale I3S project and with Telecom ParisTech LabSoC) comes from the former affiliation of E. Madelaine with Scale, dealing with formal semantics aspects of distributed systems. Concurrency is present both in distributed and embedded systems, and in fact IoT/CPS platforms are concerned with both levels. On formal models (and data-dependent bisimulation) the collaboration extends to the SPIRALS EPC (Simon Bliudze), and could also concern in the future the proposed CASH Inria team in Lyon.

On national grounds, we participate to a number of research community networks. We actively participate to the GdR SOC2 (*Groupement De Recherche "System On Chip, Systèmes embarqués et Objets Connectés"*) and the GdR GPL ("*Génie Logiciel et Programmation*"). Inside a larger CNRS initiative on Connected Objects, we coordinated the CNRS-PEPS project InS^{3}PECT (System Engineering for the design of secure services for connected objects). We are also members of the "Communauté Française de Compilation", following their activities and periodic events.

On international level, we animate specific events in the synchronous reactive language and model-based system engineering communities (with for instance the organization of Synchron 2018), and as such are frequent TPC members or even Chairs of *EmSoft, MemoCode, FDL, DATE*, FACS, and *Models* conferences, as well as the Chinese-born *TASE* conference.

In the context of the GEMOC initiative, we collaborate with international partners (mainly French, German and Canadian) both through the development of the GEMOC studio as an eclipse project but more generally on the challenges related to the use of multiple formalisms in the specification of systems. On this same subject, we participated recently to the Multi Paradigm Modeling For Cyber Physical System European COST action (MPM4CPS: http://mpm4cps.eu/).

We carry on a long-term collaboration with the Software Engineering (SEI) and Computer Science Institute of East China Normal University (Shanghai), headed by He Jifeng (also at Chinese Academy of Science). The collaboration took the form of an Inria Associated-Team, named FM4CPS, renewed once, and coupled in its last period with a LIAMA project named SACCADES supervised then by Vania

Joloboff (DR Inria). The collaboration is currently mostly funded by Chinese research programmes, and specially the (highly selective) MoE International Joint Lab of Trustworthy Software (IJLTS) for which Inria has signed in 2017 a Memorandum of Understanding with ECNU; E. Madelaine and F. Mallet are members of IJLTS advisory council. Apart from mutual visits of permanent staff leading to joint publications, this collaboration provides a rather constant flow of Chinese students towards Sophia (several International Master students and interns a year, and a PhD long-term visit on average). We also cooperate on the organization of Workshop Events and conferences.

We have also initiated in 2017 a collaboration with the University of Verona and the group of Franco Fummi, working on co-design and hybrid co-simulation at system-level. They benefit from a local funding programme from their university for the collaboration. We hosted two Master students, and hope to start PhDs on the collaboration topics (merging SystemC with FMI, stated abruptly).

We also have long-standing collaborations with the Universities of Torino and Udine on type theory for logical frameworks and object-oriented languages, and in protocols for content retrieval and overlay networks.

Kairos participated in the international partnership DNIT between UCA and University of Danang (Vietnam), mostly a student program exchanges where internships proposed on our part are reflecting our research concerns.

2.2 Industrial and contractual

Historically we have had two main kinds of contractual collaborations: some on platform-based, hardware/software co-design and High-Level Synthesis with Electronic Design Automation partners, some on Model-Based System Design with MDE partners in the embedded community. Both shared concerns for co-modeling and formal Application-Architecture Adequation approach.

On the co-modeling aspects, we have extensively collaborated in the past with Thales and CEA LIST to the definition of the MARTE UML profile for Modeling and Analysis of Embedded systems, and its relation to the wider SysML profile for System Engineering. MARTE encompasses the foundations of our vision of logical time for CPS design. SysML makes provision for “some” considerations of continuous dynamics in models, but not to the extent of considering cyber and physical models on an equal footing. Both profiles put special emphasis on the notion of mapping (physical allocation + temporal scheduling) with the joint modeling of application and computer platform models.

We also took actively part to the French PIA LEOC Clarity project, which ambitions to promote the Capella system modeling language. This design environment for Model-Based System Engineering is deployed in the large inside the various companies of Thales group, and is seriously considered in various divisions of Airbus and Areva. Many of our activities have been experimented and could be naturally embedded inside the Capella framework, where there is a strong demand for consistency between the different viewpoint of the model. For instance, we are exploring solutions, compatible with industrial requirements, to gain higher confidence in the model and the concrete implementations, through refinement/integration, concern decomposition and coordination. The Clarity project was concluded at the end of 2017, but we maintain active informal ties with many of its participants, and use the gained expertise in subsequent collaborations.

We are starting a bilateral collaboration project (Glose) with Safran, in the framework of their new

research network programme named DESIR, which associates several Inria team at the global level. The main objective is to provide a model-based solution for the integration of components into a global system view. The specific target here is to use the model as a central coordination view of the simulation models, which are interdependent while being executed by separated simulation engines. The coordination model is of central importance to drive the co-simulation process (see section 1.2). A CIFRE PhD thesis is started as part of this project.

A continuing collaboration with Safran and Airbus concerns the automatic parallelization of embedded control and monitoring code using LoPhT. This collaboration is currently supported by the ASSUME ITEA3 project (https://itea3.org/project/assume.html). Like classical compilation, the parallelization method we propose is fully automatic and scalable. Unlike classical compilation, it also takes as input non-functional requirements, e.g. resource limits or real-time requirements defined by mapping logical clocks onto physical time. The main objective is not optimization per se, but the respect of the requirements. To this end, static resource allocation and code generation algorithms perform a safe accounting of non-functional properties. Accounting starts from per-component time and memory footprint worst-case bounds, automatically obtained through calls to state-of-the-art static analysis tools. Experiments show that our method produces efficient code for large-scale, real-life avionics applications.

Similarly we have started a collaborative project named ATIPPIC in the context of an extension of IRT St Exupery in PACA**.** Project leadership is held by Thales Alenia Space, and the R&D IRT joint team is actually hosted inside Inria Sophia Centre; this time the topic is rather on software-hardware co-design, but should extend to real-time and CPS issues in the following phases. More precisely, we want to use the global system-level model to dimension the memory and the communication infrastructure of the system solution that must, at the same time, satisfy critical but sparse control commands for the navigation in space, while operating data-intensive computations for specific but evolutive scientific missions. We have recruited two technical engineers for this project, started November 2017. We should note the existence of several other projects inside this IRT which focus on neighboring topics in MBSE, such as the recently ended Moïse and Ingequip, or newly started CaPhCa project. This is a strong opportunity to maintain links with the important R&D community, mostly based around Toulouse, active inside these structures.

The FMI/FMU (https://fmi-standard.org) standardization community revolves mostly currently around Modelica industrial partners, and has so far not shown big interest for extension to cyber/event-driven aspects, not to mention formal description of model association dynamics. Still, there are multiple efforts in this direction outside the official FMI partners (with academic environments such as UC Berkeley Ptolemy, UppAal, or consortiums such as S3P, supported by the association Embedded France). We shall seek to shape up our current activities in this area, together with the University of Verona, to build new research consortium proposals. The topic is also of interest for Ansys, which acquired Esterel Technologies, and with whom we have informal talks.

We have also recently started a collaboration with Renault Software Labs for the formal modeling of rules on trajectory planning for autonomous self-driving car. The goal is to be able to monitor at runtime the planning system to detect violations of the safety rules against a set of scenarios that conform to different driving conditions. A CIFRE PhD student will start work in March 2019.

We have also recently started a collaboration with Accenture Labs on Smart Contract Languages for permissioned Blockchains & Distributed Ledgers. The goal is to focus on proposing and building extensions of existing smart contracts and/or proposing new languages for smart contracts and their

respective execution environments to overcome limitations and rigidity of existing languages and extend the current capabilities. A CIFRE PhD student will start work in 2019.

3 MILESTONES FOR SUCCESS AND RISKS

While digital models and Computer-Aided Design are becoming standard practice in other engineering discipline, Software Engineering itself is not picking up so widely on formal methods for program design. Still, the area of embedded systems, with its real-time safety-critical aspects, has been an area of intense research on this issue. This is certainly due to the specifics of reactive control software involved, which may be abstracted somehow similarly to the more general system modeling framework. From the early days the importance of timing correctness (performance, schedulability) as well as functional correctness was recognized. A wide range of formalisms were proposed, owing to process algebras, process networks, synchronous reactive languages, timed and hybrid automata, to mention the ones closest to us [1,2,3,4,17,19,21,22,23,24,25]. World-famous researchers such as E. A. Lee and A. Sangiovanni-Vincentelli (UC Berkeley), G. Berry and A. Benveniste (Inria), were strong advocates of logical time-based design, while J.Sifakis (Verimag) and K. Larsen (U. Aalborg) defended physical time in model-checking. We have had historically strong relations, and even an active role, in the realizations of this community. We keep informal relations with numerous teams across joint participation to main conferences. Still, success of formal methods sadly did not reach the audience of “mainstream” embedded designers as much as expected.
Meanwhile, system-level design environments, that were originally targeted at modeling of (possibly heterogeneous) physical systems, are progressively taking the cyber aspects of digital controllers into account (SysML/MARTE, Matlab/Simulink, Capella/ARCADIA, Ansys...), with in some cases real links to the final code of some components. But here behavioral dynamics and operational semantics descriptions remain very loose or imprecise, so that simulation and analysis results obtained at abstract model stage have little or no relevance for the final system. Here again we keep strong informal ties with prominent teams in the field, including tool architects and R&D leaders in most of the companies offering tool environments of that sort.

Our main goal is thus to enhance mainstream CPS design environments, so as to enforce formal timely operational semantics upon them, attempting to reconcile formal methods and tools on one hand, practical design flows on the other hand. Of course there is the risk of being considered as "outsiders" from either side, and the usual issues of reconciling theory with practice through experimental developments are in order here. At this point we can mention three ongoing actions that are illustrative of this trend, and representative of the current state-of-affairs at the time of writing (this is not an exhaustive list):

1. **Heterogeneous MoCCs and Model-Based System Engineering**. We share participation with HyCOMES and DiVerSE EPIs on a common case-study of drone model in the Glose collaboration proposed by Safran, which combines the semantic and syntactic DSML aspects of cyber-physical combinations. We also have frequent informal exchanges with Parkas and Tea members on hybrid synchronous models; we keep strong contacts with the academic and industrial users of Capella, AADL, and SysML/MARTE methodologies, in particular through our renewed involvement in IRT Saint-Exupery. We also keep informal links with Ansys, now owner of SCADE but also involved in MBSE and heterogeneous simulation.
2. **Platform-Based Design and relation to code**. Following some recently terminated long-term collaborations (such as CIM PACA Design Platform or ANR HOPE/HelP), we are keeping strong

ties with former partners (on SystemC language and HW/SW co-design, on Manycore Systems-on-Chip and On-chip Networks). From the ITEA3 Assume and ES3CAP project and other links we have contacts with the parallel compilation community.

3. **Non-Functional Property checking and IoT**. We are starting to consider Safety Analysis (in the sense of Fault Tolerance) and Security property modeling in connection with logical time to express associated temporal conditions, in our collaborations inside ATIPPIC and SPAI respectively.

As all research teams we have set ourselves classical objectives in term of (co)-publications, collaborative contracts, software dissemination, and student supervision, with the intention to raise joint work between team members as well as increasing our research scope. We may try to state a few perspectives for future goals (again, not an exhaustive list, according to prediction):

Short term perspectives (2 years)

One main objective of Kairos is to keep (and even expand) a strong link between a formal approach based on theoretical computer science, and a practical system design methodology covered at most levels, in the specific domain of embedded and cyber-physical systems. Practical measures of success will stem from actual proof-of-concept realizations on use-case applications. The efforts planned into both the IRT Saint-Exupery ATIPPIC project and the Safran DESIR Glose collaboration should provide such opportunities to meet users requests; this should greatly benefit from the larger interest inside IRT Saint-Exupery for such MBSE developments, which is also greatly recognized by industrial partners from IRT SystemX

This short term action should mainly result in combining our expertise in MBSE and in AAA by integrating into the GeMoC studio the co-design and code generation capabilities provided by LoPhT. The case study in the domain of Space brings new non-functional properties (radiation model, load, hardness of chips) that were not considered so far and that very much characteristics of the challenges of designing CPS.

On the AAA side, we target formalization of new mapping constraints originated from mapping problems relevant both to the scientific community and to the industrial needs (like the need to use less time-predictable multi-cores based on ARM or POWER architectures). This should be defined through joint participation in the PIA ES3CAP project, the IRT Saint-Exupéry CAPHCA project, and a direct collaboration with Airbus.

Mid-term perspectives (6 years)

The challenge of integrating models mixing discrete and continuous time models into MBSE should be addressed along this collaboration with SAFRAN in Glose. It particular it demands to extend our logical time models with constructs to combine discrete/continuous time, to take into account data-dependent control and to demonstrate that model-based approaches can be used to control the co-simulation between the set of tools involved, while possibly allow the generation of sound glue code (like master algorithms).

Beyond these relations with large industrial groups on a rather institutional R&D collaborative mode, that may be naturally established, there are more uncertain challenges in keeping track with CAD software houses and tool providers. This is of utter importance to us, as we do not intent so much to produce our own stand-alone tool sets, as to provide add-ons and supplement some new features into existing design frameworks (such as Capella, SysML/MARTE, AADL). Nevertheless, in several other circumstances we may also allow ourselves independent software production.

Another goal yet-to-be-met would be to extend the range of potential users, from classical MBSE

firms such as Thales, Airbus, Safran, TAS, to partners in the automotive domain, which are increasing their presence in Sophia Antipolis area, as well as to IoT major players such as Orange and SAP and innovative SMEs. The recent collaboration with Renault Software Lab demands to adapt our logical time abstractions and our tool TimeSquare to integrate mobility and dynamicity aspects.

There are important remaining issues regarding the coupling of "traditional"l MBSE with "language-based" approaches, that are reflected in in the embedded system design community (e.g. Lustre/SCADE, SystemJ), in the parallel compilation community (polyhedral compilation, C11/pthreads code generation), and in the classical compilation community (e.g. gcc/CompCert). To ensure predictability and efficiency, integration between methods and tools of the various fields is needed. A key issue here is the identification of exchange formalisms/languages with the appropriate expressivity to allow end-to-end formal integration and reasoning on both correctness and efficiency.

Long-term perspectives (> 6 years)

While it is always difficult to state long term objectives, we could reasonably argue that the research to be done in the Kairos team could lead to an experimental open-source modeling platform for the design and verification of cyber-physical systems based on logical time abstractions with the coordination of several ad-hoc formal models of computations and domain-specific languages. The formal semantics of the coordination model would be:

- the base for the coordinated execution of state-of-the-art simulation engines from the different subdomains ;
- used to generate compiler-friendly, therefore efficient, control source code (in C++) for the safe integration of the different parts of the system;
- used in the refinement process for checking the behavioral equivalence between the models at different refinement levels and eventually the candidate implementation.

We also expect to contribute logical time constructs into mainframe programming languages like Java or go, by designing adequate new constructs and types as logical-time extensions of current frameworks based on Type Theory.

4 Software tools

As a rule of thumb, our software development efforts are more triggered at proof-of-concept than full-fledge software products. Still, the concern for compatibility and capacity to be used as plug-ins or add-ons in existing design environment remains constant. The main reasons for this situation may be that efforts for large-scale industrialization of synchronous languages such as Scade and Esterel have already been conducted in e.g the past, and also that MBSE design flows are largely deployed in many forms in major industrial partners (Thales, Airbus, Safran, ST Microelectronics), so that acceptance of advanced techniques require more sophisticated strategies. We have witnessed in the past concrete successes in influence, such as the adoption of SyncCharts in Scade, the inclusion of Logical Time features in commercial UML modelers such as Obeo Designer or MagicDraw (now owned by Dassault Systems), or more recently the MARTE view inside Capella developed internally at Thales. We keep frequent contacts with MBSE commercial tool builders in different settings, such as the ECLIPSE Polarsys project.

We now give a list of significant developments:

TimeSquare. APP IDDN.FR.001.170007.000.s.P.2009.000.10600 [http://timesquare.inria.fr]
We have developed the TimeSquare simulation and analysis environment for the CCSL formalism,

which is in act meant to be an experimental lab for the handling of logical time constraints in simulation and schedulability analysis. This module is not primarily meant to be used stand-alone, but rather to be integrated into larger frameworks (somehow like (SMT) solvers, for instance). It is promoted inside many collaborative partnerships (ARTEMIS Presto, ANR GeMoC, PIA Léoc Clarity, bilateral partnership with Thales in its ARCADIA/CAPELLA solution).

Gemoc studio. [http://gemoc.org/]
was co-developed inside the ANR GeMoC project and as part of the GeMoC initiative. It allows the definition of multiple interacting Models of Computation, with formal description of their association constraints (somehow in the fashion of UC Berkeley Ptolemy, but with formal logical time constraints specifying the operations of directors, instead of Java code). While Gemoc Studio results from collaborations, there are two clear contributions from Aoste (MoCCML and BCoOL) that will be essential for to address Cyber Physical Systems modeling.

- MoCCML APP IDDN.FR.001.470022.000.S.P.2017.000.10600 (Model of Concurrency and Communication Modeling Language) [DDTC15]. It is a formal language to capture the concurrent and communication part of the semantics of a language.

- BCOoL APP IDDN.FR.001.470021.000.S.P.2017.000.10600 (Behavioral Coordination Operator Language) [VLDC15]. BCOoL is a language dedicated to the specification of coordination patterns between heterogeneous languages. It comes with a tool chain allowing the generation of the coordination given a BCOoL operator and specific models.

GeMoC is at the center of the Glose collaboration with Safran, part of the global R&D programme DESIR between academics and Safran. It is also at the heart of transfer discussions with SATT Sud-Est towards local SMEs.

VerCors. [https://team.inria.fr/scale/software/vercors/vcev4-download/]
This is a specification and verification platform, inherited from the works of the Oasis, Scale, and Aoste EPIs. VerCors front-end is based on MDE technology, using the Sirius platform for the graphical editors and transformation to semantic models. The verification part offers links with the CADP toolset for model-checking. It has been used in several places for teaching, and in the OpenCloudware FUI project for developing industrial inspired use-cases.
We are now building our own prototype algorithms within VerCors, to implement and validate the work described in section 1.3.

LoPhT. APP IDDN.FR.001.090043.000.S.P.2016.000.10600
Lopht is a system-level compiler for embedded systems. It fully automates the implementation of 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-independent description of the functionality and of the non-functional requirements (e.g. real-time, partitioning). This is provided under the form of a multi-clock 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.

5 Teaching and education

Due to the strong presence of UNS professors and assistant-professors in the team we are actively involved in local teaching curricula, some of them connected with our research focus. We are involved in the International Track of the Master in Computer Science (both years, M1 and M2 UbiNet). Frederic Mallet is the scientific coordinator of the M1.
We built a new proposal for an integrated international Master programme supported by UCA, named ROCC (for "Réseaux, Objets connectés: du Capteur au Cloud"), in which we play a central role. The goal is to bring together competencies from various actors, from the design of antenna, medium range communication protocols, and system-on-chips to widely distributed systems in the cloud, including communication and network infrastructures. This new master will bring together the two current degrees of UNS in Computer Science and Electrical Engineering along with researchers from Inria and CNRS. It would be a central piece in the teaching part of the upcoming EUR (Ecole Universitaire de Recherche) proposal carried by the RISE Academy of the IDEX UCA Jedi.
Following our involvement with Chinese partners, the International Master in Computer Science of UCA is now officially double degree with ECNU Shanghai.

6 EXPERTISE OF THE TEAM MEMBERS

Robert de Simone is a Research Director with Inria. He was the Scientific leader of the former Aoste EPI, dedicated to Modeling and Analysis of Real-Time Embedded Systems. Co-hosted in Paris and Sophia Inria centers, and joint with the UMR I3S from CNRS/University of Nice Sophia Antipolis. He has worked on process algebras, synchronous languages and various topics on expressiveness and operational semantics foundations of concurrent Models of Computation and Communication, including practical aspects involved in automatic verification and model-checking. He has put special efforts in the exposure of formal methods to general-purpose Model-Driven Engineering for Embedded and Real-Time systems, with contributions to the OMG MARTE UML profile and the CCSL formalism for logical time constraints. He has been involved in dedicated modeling for Hardware/Software co-design, mainly as part of the CIM PACA regional collaboration initiative with local major industrial partners. He has written numerous articles, and supervised over a dozen PhD students.

Frederic Mallet is a Professor of Computer Science at Université Nice Sophia Antipolis and deputy director of the I3S Laboratory. He is in charge of the International Track of the Master in Computer Science and is carrying the RoCC degree as part of UCA Jedi's Master Programme and EUR DS4H project. His research interests lie in the definition of sound polychronous models of time for the design of architecture-dependent software systems with potential safety issues. Since 2006, he has been involved in the definition of MARTE Time Model, of its formal companion language CCSL, and of its positioning compared to other emerging industry standards like SysML and AADL. Addressing Cyber-Physical Systems brings new exciting challenges to be addressed by Kairos team, among which, 1) the definition of *dense* logical clocks to model artifacts obeying continuous physical laws, 2) the characterization of the *uncertain* behaviors of the surrounding environment and of the interconnect, 3) the integration of *security* concerns required by the open nature of CPS. http://www.i3s.unice.fr/~fmallet/

Julien Deantoni received a Ph.D. degree in Computer Science from the INSA–Lyon engineering

school, France in 2007. His Ph.D was realized in the embedded systems team of the CITI laboratory from 2004 to 2007 where he successfully participated to international robotic challenges to highlight the benefits of model driven analysis. After being a postdoctoral fellow (funded by the European SPEEDS project) in the Triskell EPI at INRIA Rennes, he is currently an associate professor at University Nice Sophia Antipolis and a member of the Kairos team (UCA, INRIA & I3S). His research interest concerns the use and the definition of formal methods and tools dedicated to model driven engineering. More information is available at http://www.i3s.unice.fr/~deantoni/

Luigi Liquori is a Research Director at Inria. MS 1990 Udine University, Ph.D. 1996 University of Turin, HdR. 2007 Institut National Polytechnique de Lorraine, served as a Lecturer at the Ecole Nationale des Mines de Nancy from 1999 to 2001. Luigi Liquori research's fields range from lambda-calculus, type theory, logical frameworks, to semantics of object oriented programming languages, until foundations of overlay networks.
http://www-sop.inria.fr/members/Luigi.Liquori/

Eric Madelaine is a Research scientist with INRIA. He has been Scientific Leader of the former Oasis EPC (joint team with the UMR I3S from CNRS/University Nice Sophia Antipolis), dedicated to semantic and programming models, verification methods, and execution platforms for distributed systems. He has worked on process algebras, semantics of asynchronous languages and distributed software components, tools for high-level design and formal verification of such systems. He has a strong involvement in the research community on formal aspects of software languages, and in particular distributed systems.

Sid Touati is a Professor at the computer science department of the University Nice Sophia Antipolis. He is currently co-responsible of the COMRED department in the I3S laboratory, in which the Kairos team belongs. He has many teaching responsibilities at various levels of licence and master of computer science. He is an expert in advanced code optimisation and compilation, code performance analysis and evaluation for HPC and embedded processors, statistical studies regarding performance comparison, etc. He is interested in code performance variability: how to analyse it with parametric and non-parametric statistics, how to model it, how to understand it, how to reduce it.
For more details: http://www-sop.inria.fr/members/Sid.Touati/

Marie-Agnès Peraldi-Frati is an Associate Professor in computer science at the University Nice Sophia Antipolis. After a PHD in automatic and signal processing, she had a postdoctoral position at the Swiss Federal Institute of Technology in Lausanne. She is a member of the Kairos team and its research field is related to the modeling and analysis of non-functional constraints on objects/services with the perspective of mixing real-time constraints (performance, power-consumption) with constraints more specific from the application domains (home care services, Daily Living monitoring) such as privacy of data , non-intrusive equipment. She taught in different degrees of the University Nice Sophia Antipolis. More information can be found in http://www.i3s.unice.fr/~map/

Dumitru Potop-Butucaru received his Ph.D. degree from Ecole des Mines de Paris, in 2002, and his Habilitation to conduct research (HDR) from University Pierre et Marie Curie (Paris 6) in 2015. He joined the AOSTE EPC in 2005. Continuing a successful history of academic and industrial collaborations, his current research introduces and promotes the concept of Real-Time Systems Compilation (https://hal.inria.fr/tel-01264021). By analogy with classical compilation, real-time

systems compilation consists in the fully automatic and scalable construction of running, correct-by-construction implementations from functional and non-functional specifications of embedded control systems.
Personal web site: https://who.rocq.inria.fr/Dumitru.Potop_Butucaru/

7 SELECTED BIBLIOGRAPHY

The following list covers only some publications by team members most relevant to Kairos proposal.

[AM08] Charles André, Frédéric Mallet: Clock Constraints in UML/MARTE CCSL. Inria Research Report RR6540, INRIA. 2008.

[And09] Charles André, Syntax and Semantics of the Clock Constraint Specification Language (CCSL), Inria Research Report RR6925, 2009.

[BCEL15] Boudjennah, Christophe ; Combemale, Benoit ; Exertier, Daniel ; Lacrampe, Stéphane ; Peraldi-Frati, Marie-Agnès: CLARITY: Open-Sourcing the Model-Based Systems Engineering Solution Capella. In: *Second Workshop on Open Source Software for Model Driven Engineering*, 2015

[CDBF14] Combemale, Benoit ; DeAntoni, Julien ; Baudry, Benoit ; France, Robert B. ; Jezequel, Jean-Marc ; Gray, Jeff: Globalizing Modeling Languages. In: *Computer* Bd. 47 (2014), Nr. 6, S. 68–71

[CDLM13] Combemale, Benoît ; De Antoni, Julien ; Larsen, Matias Vara ; Mallet, Frédéric ; Barais, Olivier ; Baudry, Benoit ; France, Robert B.: Reifying Concurrency for Executable Metamodeling. In: *Lecture Notes in Computer Science*, 2013, S. 365–384

[CDPS14] Carle, Thomas ; Djemal, Manel ; Potop-Butucaru, Dumitru ; de Simone, Robert ; Zhang, Zhen: Static Mapping of Real-Time Applications onto Massively Parallel Processor Arrays. In: *2014 14th International Conference on Application of Concurrency to System Design*, 2014

[CeDS16] Centomo, Stefano ; Deantoni, Julien ; de Simone, Robert: Using SystemC Cyber Models in an FMI Co-Simulation Environment: Results and Proposed FMI Enhancements. In: *2016 Euromicro Conference on Digital System Design (DSD)*, 2016

[CGLL13] Ciancaglini, Vincenzo ; Gaeta, Rossano ; Loti, Riccardo ; Liquori, Luigi: Interconnection of Large Scale Unstructured P2P Networks: Modeling and Analysis. In: *20th International Conference on Analytical and Stochastic Modelling Techniques and Applications (ASMTA), Lecture Notes in Computer Science*, 2013, S. 183–197

[DDTC15] Deantoni, Julien ; Diallo, Issa Papa ; Teodorov, Ciprian ; Champeau, Joel ; Combemale, Benoit: Towards a Meta-Language for the Concurrency Concern in DSLs. In: *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2015*, 2015

[Dean16] Deantoni, Julien: Modeling the Behavioral Semantics of Heterogeneous Languages and their Coordination. In: *2016 Architecture-Centric Virtual Integration (ACVI)*, 2016

[DHJM18] Du, Dehui ; Huang, Ping ; Jiang, Kaiqiang ; Mallet, Frédéric: pCSSL: a Stochastic Extension to MARTE/CCSL for Modeling Uncertainty in Cyber Physical Systems. In: *Science of Computer Programming* (2018)

[DLLS16] Dougherty, Daniel J. ; de'Liguoro, Ugo ; Liquori, Luigi ; Stolze, Claude: A Realizability Interpretation for Intersection and Union Types. In: *14th Asian Symposium on Programming Languages and Systems (APLAS), Lecture Notes in Computer Science*, 2016, S. 187–205

[GDMM14] Glitia, Calin ; DeAntoni, Julien ; Mallet, Frédéric ; Millo, Jean-Vivien ; Boulet, Pierre ; Gamatié, Abdoulaye: Progressive and explicit refinement of scheduling for multidimensional data-flow applications using UML MARTE. In: *Design Automation for Embedded Systems* Bd. 19 (2014), Nr. 1-2, S. 1–33

[GKCP15] Gorcitz, Raul ; Kofman, Emilien ; Carle, Thomas ; Potop-Butucaru, Dumitru ; de Simone, Robert: On the Scalability of Constraint Solving for Static/Off-Line Real-Time Scheduling. In: *Lecture Notes in Computer Science*, 2015, S. 108–123

[GoDM13] Gomez, Carlos ; DeAntoni, Julien ; Mallet, Frederic: Power consumption analysis using multi-view modeling. In: *2013 23rd International Workshop on Power and Timing Modeling, Optimization and Simulation (PATMOS)*, 2013

[HeMZ16] Henrio, Ludovic ; Madelaine, Eric ; Zhang, Min: A Theory for the Composition of Concurrent Processes. In: *Lecture Notes in Computer Science*, 2016, S. 175–194

[HLCM13] Hoang, Giang Ngo ; Liquori, Luigi ; Ciancaglini, Vincenzo ; Maksimovic, Petar ; Chan, Hung Nguyen: A backward-compatible protocol for inter-routing over heterogeneous overlay networks. In: *Proceedings of the 28th Annual ACM Symposium on Applied Computing - SAC '13*, 2013

[HLMS17] Honsell, Furio ; Liquori, Luigi ; Maksimovic, Petar ; Scagnetto, Ivan: 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* Bd. 13 (2017)

[HLMS18] Honsell, Furio ; Liquori, Luigi ; Maksimović, Petar ; Scagnetto, Ivan: Plugging-in proof development environments using Locks in LF. In: *Mathematical Structures in Computer Science* (2018), S. 1–28

[HLSL13] Honsell, Furio ; Lenisa, Marina ; Scagnetto, Ivan ; Liquori, Luigi ; Maksimovic, Petar: An open logical framework. In: *Journal of Logic and Computation* Bd. 26 (2013), Nr. 1, S. 293–335

[KhMR17] Khan, Aamir M. ; Mallet, Frédéric ; Rashid, Muhammad: A framework to specify system requirements using natural interpretation of UML/MARTE diagrams. In: *Software & Systems Modeling* (2017)

[KoSi16] Kofman, Emilien ; de Simone, Robert: A formal approach to the mapping of tasks on an heterogeneous multicore, energy-aware architecture. In: *2016 ACM/IEEE International Conference on Formal Methods and Models for System Design (MEMOCODE)*, 2016

[LDPQ18] Liboni, Giovanni ; Deantoni, Julien ; Portaluri, Antonio ; Quaglia, Davide ; de Simone, Robert: Beyond Time-Triggered Co-simulation of Cyber-Physical Systems for Performance and Accuracy Improvements. In: *Proceedings of the Rapido'18 Workshop on Rapid Simulation and Performance Evaluation: Methods and Tools - RAPIDO '18*, 2018

[LiSe17] Liquori, Luigi ; Sereno, Matteo: LogNet: Extending Internet with a Network Aware Discovery Service, https://hal.inria.fr/hal-01895452v1, (2017)

[LiSt17] Liquori, Luigi ; Stolze, Claude: A Decidable Subtyping Logic for Intersection and Union

Types. In: *International Conference on Topics in Theoretical Computer Science (TTCS), Lecture Notes in Computer Science*, 2017, S. 74–90

[LLHM13] Liu, Jing ; Liu, Ziwei ; He, Jifeng ; Mallet, Frédéric ; Ding, Zuohua: Hybrid MARTE statecharts. In: *Frontiers of Computer Science* Bd. 7 (2013), Nr. 1, S. 95–108

[Mall15] Mallet, Frédéric: MARTE/CCSL for Modeling Cyber-Physical Systems. In: *Formal Modeling and Verification of Cyber-Physical Systems*, 2015, S. 26–49

[MaMD13] Mallet, Frédéric ; Millo, Jean-Vivien ; De Simone, Robert: Safe CCSL specifications and marked graphs - IEEE Conference Publication. In: *11th IEEE/ACM International Conference on Formal Methods and Models for Codesign (MEMOCODE)* : IEEE, 2013

[MaSi15] Mallet, Frédéric ; de Simone, Robert: Correctness issues on MARTE/CCSL constraints. In: *Science of Computer Programming* Bd. 106 (2015), S. 78–92

[MaVH17] Mallet, Frédéric ; Villar, Eugenio ; Herrera, Fernando: MARTE for CPS and CPSoS. In: *Cyber-Physical System Design from an Architecture Analysis Viewpoint*, 2017, S. 81–108

[MaZh16] Madelaine, Eric ; Zhang, Min: Towards a bisimulation theory for open synchronized networks of automata. In: *Science China. Information Sciences* Bd. 59 (2016), Nr. 5

[MiKD15] Millo, Jean-Vivien ; Kofman, Emilien ; De Simone, Robert: Modeling and Analyzing Dataflow Applications on NoC-Based Many-Core Architectures. In: *ACM Transactions on Embedded Computing Systems* Bd. 14 (2015), Nr. 3, S. 1–25

[MDAS10] F. Mallet, J. DeAntoni, C. André, R. de Simone: The clock constraint specification language for building timed causality models - Application to synchronous data flow graphs. ISSE 6(1-2): 99-106 (2010)

[MiSi12] Millo, Jean-Vivien ; de Simone, Robert: Periodic scheduling of marked graphs using balanced binary words. In: *Theoretical computer science*. 458 (2012), S. 113–130

[MPDS14] Mallet, Frédéric ; Peraldi-Frati, Marie-Agnès ; Deantoni, Julien ; de Simone, Robert: UML MARTE Time Model and Its Clock Constraint Specification Language. In: *Advances in Systems Analysis, Software Engineering, and High Performance Computing*, 2014, S. 29–51

[NgLN14] Ngo, Hoang Giang ; Liquori, Luigi ; Nguyen, Chan Hung: Backward-Compatible Cooperation of Heterogeneous P2P Systems. In: *15th International Conference on Distributed Computing and Networking (ICDCN), Lecture Notes in Computer*, 2014, S. 287–301

[PGDN12] Peraldi-Frati, Marie-Agnes ; Goknil, Arda ; DeAntoni, Julien ; Nordlander, Johan: A timing model for specifying multi clock automotive systems: The timing augmented description language v2. In: *International Conference on Engineering of Complex Computer Systems*, 2012

[Potop2015] Dumitru Potop-Butucaru: Real-Time Systems Compilation, HDR thesis, 2015.

[PST09] Potop-Butucaru D., De Simone, R., Talpin, J.-P. : "The synchronous hypothesis and polychronous languages" Chapter in Networked Embedded Systems Handbook, CRC Press, 2009

[QBMZ18] Qin, Xudong ; Bliudze, Simon ; Madelaine, Eric ; Zhang, Min: Using SMT engine to generate Symbolic Automata. In: *18th International Workshop on Automated Verification of Critical Systems*, 2018

[PSST11] D. Potop-Butucaru, Y. Sorel, R. de Simone, and J-P. Talpin: From Concurrent Multi-

clock Programs to Deterministic Asynchronous Implementations. Fundam. Inf. 108, 1-2 (January 2011), 91-118.

[SLHS17] Stolze, Claude ; Liquori, Luigi ; Honsell, Furio ; Scagnetto, Ivan: Towards a Logical Framework with Intersection and Union Types. In: *Proceedings of the Workshop on Logical Frameworks and Meta-Languages: Theory and Practice - LFMTP*, 2017

[HLSS18] Honsell, Furio ; Liquori, Luigi ; Scagnetto, Ivan ; Stolze, Claude : The Delta Framework. In: *38th {IARCS} Annual Conference on Foundations of Software Technology and Theoretical Computer Science - FSTTCS, 2018. Ahmedabad, India*

[ToDi14] Touati, Sid ; de Dinechin, Benoit Dupont: *Advanced Backend Code Optimization,* ISBN 978-1-84821-538-2, ISTE edition, Wiley Publisher, 2014

[VLDC15] Vara Larsen, Matias Ezequiel ; Larsen, Matias Ezequiel Vara ; DeAntoni, Julien ; Combemale, Benoit ; Mallet, Frederic: A Behavioral Coordination Operator Language (BCOoL). In: *2015 ACM/IEEE 18th International Conference on Model Driven Engineering Languages and Systems (MODELS)*, 2015

[WoTo16] Worms, Julien ; Touati, Sid: Going beyond Mean and Median Programs Performances. In: *2016 IEEE 10th International Symposium on Embedded Multicore/Many-core Systems-on-Chip (MCSOC)*, 2016

[WoTo17] Worms, Julien ; Touati, Sid: Modelling Program's Performance with Gaussian Mixtures for Parametric Statistics. In: *IEEE Transactions on Multi-Scale Computing Systems* (2017), S. 1–1

[YLDM13] Yin, Ling ; Liu, Jing ; Ding, Zuohua ; Mallet, Frederic ; de Simone, Robert: Schedulability Analysis with CCSL Specifications. In: *2013 20th Asia-Pacific Software Engineering Conference (APSEC)*, 2013

[ZhDM18] Zhang, Min ; Dai, Feng ; Mallet, Frédéric: Periodic scheduling for MARTE/CCSL: Theory and practice. In: *Science of Computer Programming* Bd. 154 (2018), S. 42–60

8 EXTERNAL REFERENCES

1. Derler P, Lee EA, Vincentelli AS. Modeling Cyber–Physical Systems. Proc IEEE. 2012;100: 13–28. doi:10.1109/jproc.2011.2160929

2. Alur R. Principles of Cyber-Physical Systems [Internet]. MIT Press; 2015. Available electronically: https://books.google.com/books/about/Principles_of_Cyber_Physical_Systems.html?hl=&id=E7QICAAAQBAJ

3. Lee EA, Rabaey J, Hartmann B, Kubiatowicz J, Pister K, Sangiovanni-Vincentelli A, et al. The Swarm at the Edge of the Cloud. IEEE Des Test Comput. 2014;31: 8–20. doi:10.1109/mdat.2014.2314600

4. Lee EA, Seshia SA. Introduction to Embedded Systems: A Cyber-Physical Systems Approach [Internet]. MIT Press; 2016. Available: https://books.google.com/books/about/Introduction_to_Embedded_Systems.html?hl=&id=chPiDQAAQBAJ

5. Lamport L. Time, clocks, and the ordering of events in a distributed system. Commun ACM.

1978;21: 558–565. doi:10.1145/359545.359563

6. Fidge C. Logical time in distributed computing systems. Computer . 1991;24: 28–33. doi:10.1109/2.84874

7. ISO/IEC/IEEE 42010-2011 Standard: Systems and software engineering -- Architecture description. doi:10.1109/ieeestd.2011.6130559

8. Combemale B, DeAntoni J, Baudry B, France RB, Jezequel J-M, Gray J. Globalizing Modeling Languages. Computer . 2014;47: 68–71. doi:10.1109/mc.2014.147

9. Balasubramanian D, Păsăreanu CS, Karsai G, Lowry MR. Polyglot: Systematic Analysis for Multiple Statechart Formalisms. Lecture Notes in Computer Science. 2013. pp. 523–529. doi:10.1007/978-3-642-36742-7_36

10. Cheng BHC, Combemale B, France RB, Jézéquel J-M, Rumpe B. On the Globalization of Domain-Specific Languages. Lecture Notes in Computer Science. 2015. pp. 1–6. doi:10.1007/978-3-319-26172-0_1

11. Chauvel F, Jézéquel J-M. Code Generation from UML Models with Semantic Variation Points. Lecture Notes in Computer Science. 2005. pp. 54–68. doi:10.1007/11557432_5

12. Ciccozzi F, Cicchetti A, Sjodin M. Exploiting UML Semantic Variation Points to Generate Explicit Component Interconnections in Complex Systems. 2013 10th International Conference on Information Technology: New Generations. 2013. doi:10.1109/itng.2013.37

13. Grönniger H, Reiß D, Rumpe B. Towards a Semantics of Activity Diagrams with Semantic Variation Points. Lecture Notes in Computer Science. 2010. pp. 331–345. doi:10.1007/978-3-642-16145-2_23

14. Lee EA, Sangiovanni-Vincentelli A. A framework for comparing models of computation. IEEE Trans Comput Aided Des Integr Circuits Syst. 1998;17: 1217–1229. doi:10.1109/43.736561

15. Bourke T, Pouzet M. Zélus. Proceedings of the 16th international conference on Hybrid systems: computation and control - HSCC '13. 2013. doi:10.1145/2461328.2461348

16. Talpin J-P, Jouvelot P, Shukla SK. Liquid Clocks-Refinement Types for Time-Dependent Stream Functions [Internet]. INRIA Rennes-Bretagne Atlantique; INRIA. 2015. Available: https://hal.inria.fr/hal-01166350/

17. Eker J, Janneck JW, Lee EA, Liu J, Liu X, Ludvig J, et al. Taming heterogeneity - the Ptolemy approach. Proc IEEE. 2003;91: 127–144. doi:10.1109/jproc.2002.805829

18. Balasubramanian, Daniel, Corina S. Păsăreanu, Michael W. Whalen, Gábor Karsai, and Michael Lowry. "Polyglot: modeling and analysis for multiple statechart formalisms." In *Proceedings of the 2011 International Symposium on Software Testing and Analysis*, pp. 45-55. ACM, 2011.

19. Hartmanns A., Hermanns H., "In the quantitative automata zoo", Science of Computer Programming, 112 (2015).

20. Gérard Berry, "L'informatique du temps et des évènements", Leçon Inaugurale du Cours au Collège de France 2012-2013, http://books.openedition.org/cdf/3300

21. Joseph Sifakis, "System Design in the Era of IoT - Meeting the Autonomy Challenge", Invited Paper, Workshop on Methods and Tools for Rigorous System Design (MeTRiD 2018).

22. “Principles of Modeling”, Essays Dedicated to Edward A. Lee on the Occasion of His 60th Birthday, Springer International Publishing, 2018, DOI:10.1007/978-3-319-95246-8

23. “20 Years of Real Real Time Model Validation”, Kim Guldstrand Larsen, Florian Lorber, Brian Nielsen, Proceedings of 22th Congress on Formal Methods (FM 2018), 2018.

24. “Embedded Systems Handbook”, ed. Richard Zurawski, CRC Taylor $ Francis publisher, 2006.

25. Albert Benveniste, Benoît Caillaud, Dejan Nickovic, Roberto Passerone, Jean-Baptiste Raclet, Philipp Reinkemeier, Alberto L. Sangiovanni-Vincentelli, Werner Damm, Thomas A. Henzinger, Kim G. Larsen: “Contracts for System Design”. Foundations and Trends in Electronic Design Automation (2018)

26. Nuzzo P, Sangiovanni-Vincentelli AL, Bresolin D, Geretti L, Villa T. A Platform-Based Design Methodology With Contracts and Related Tools for the Design of Cyber-Physical Systems. Proc IEEE;103: 2104–2132. (2015)

27. Akkaya I, Derler P, Emoto S, Lee EA. Systems Engineering for Industrial Cyber–Physical Systems Using Aspects. Proc IEEE;104: 997–1012. (2016)

28. A. Benveniste ; P. Caspi ; S.A. Edwards ; N. Halbwachs ; P. Le Guernic ; R. de Simone: The synchronous languages 12 years later, Proceedings of the IEEE Volume: 91 , Issue: 1 , Jan 2003.

29. A.Malik, Z. Salcic, P Roop, A. Girault: “SystemJ: A GALS language for system level design”, Computer Languages, Systems & Structures Volume 36, Issue 4, December 2010, Pages 317-34

9 ACRONYMS

AAA - Application/Architecture Adaptation - http://www.syndex.or/

AADL - Architecture Analysis & Description Language - http://www.aadl.info/

ATP - Automatic Theorem-Prover

CCSL - The Clock Constraint Specification Language

CPS - Cyber Physical Systems: http://cyberphysicalsystems.org/

DS(M)L - Domain Specific (Modeling) Languages

FMI/FMU - Functional Mockup Interface / Functional Mockup Unit - http://fmi-standard.org

IoT - Internet of Things

LF - Edinburgh Logical Framework

MARTE - UML Profile for Modeling and Analysis of Real-Time and Embedded Systems - https://www.omg.org/omgmarte/

MBSE - Model-Based System Engineering [27]

MoCC - Models of Computation and Communication

NFP - Non-Functional Properties

PBD - Platform-Based Design [26]

SDF - Synchronous Data Flow

SMT - Satisfiability Modulo Theory

SOS - Structural Operational Semantics

SysML - UML Profile for Systems Engineering - http://www.sysml.org

WCET - Worst-Case Execution Time