Activity Report 2018

# Team AOSTE2

## Models and methods of analysis and optimization for systems with real-time and embedded contraints

RESEARCH CENTER
**Paris**

THEME
**Embedded and Real-time Systems**

# Table of contents

# Team AOSTE2

*Creation of the Team: 2017 January 01*

**Keywords:**

**Computer Science and Digital Science:**

A1.3. - Distributed Systems
A1.5.2. - Communicating systems
A2.1.1. - Semantics of programming languages
A2.1.9. - Synchronous languages
A2.1.10. - Domain-specific languages
A2.2.4. - Parallel architectures
A2.2.5. - Run-time systems
A2.3. - Embedded and cyber-physical systems
A2.3.1. - Embedded systems
A2.3.2. - Cyber-physical systems
A2.3.3. - Real-time systems
A2.4.1. - Analysis
A2.4.3. - Proofs
A8.2. - Optimization

**Other Research Topics and Application Domains:**

B5.2. - Design and manufacturing
B5.2.1. - Road vehicles
B5.2.2. - Railway
B5.2.3. - Aviation
B5.2.4. - Aerospace
B6.6. - Embedded systems

# 1. Team, Visitors, External Collaborators

**Research Scientists**

Yves Sorel [Team leader, Inria, Senior Researcher]
Liliana Cucu [Inria, Researcher, HDR]
Robert Davis [University of York UK, Chair]
Dumitru Potop Butucaru [Inria, Researcher, HDR]

**PhD Students**

Slim Ben Amor [Inria]
Keryan Didier [Inria]
Evariste Ntaryamira [Inria, Embassy of France at Burundi]
Salah Eddine Saidi [IFPEN, until Mar 2018]
Walid Talaboulma [Inria]

**Technical staff**

Adriana Gogonel [Inria, until Aug 2018]
Fatma Jebali [Inria]
Cristian Maxim [Inria, until Aug 2018]

Mehdi Mezouak [Inria]

**Intern**

Khalid Saadi [Inria, from Apr 2018 until Sep 2018]

**Administrative Assistant**

Christine Anocq [Inria]

**External Collaborator**

Adriana Gogonel [StatInf, from September 2018]

# 2. Overall Objectives

## 2.1. Overall Objectives

The recent advances in merging different technologies and engineering domains has led to the emergence of Cyber-Physical Systems (CPS). In such systems, embedded computers interact with, and control physical processes. These embedded computers (cyber) may communicate from a tightly coupled way, for example through a serial CAN bus in the automotive domain or through an AFDX bus in the avionics domain to control engine(s) or brakes (physics), to a loosely coupled way for example through the internet network to offer multimedia services or data-base accesses. Because of the heterogeneity of the involved components (multi-physics, sensors, actuators, embedded computers), CPS may feature very complex design and implementation phases as well as complex computer platforms (multi/manycore, multiprocessor, distributed and parallel computers), ever raising the need for effective approaches in order to build reliable systems.

Most of these CPS are time sensitive, i.e. time is a crucial issue which must be carefully mastered, that yet increases their complexity. Mastering time in such CPS is the major objective of the team. Due to their heterogeneous nature, the different components may have different levels of criticality, e.g. engine and brakes have a higher criticality level than multimedia services, which increase the difficulty in the design and implementation phases since lower criticality parts must not interfere with higher criticality parts. In the team we mainly address mixed-criticality issues in term of software safety. However, we started to take into account, in addition, security issues (cyber attacks).

The members of the team beeing involved for a long time in *synchronous languages*, we address the functional specification of CPS with models compliant with the semantics of these languages. Theses models are basicaly graphs and more specifically “clocked graphs” that model data dependences beetween functions as well as “logical clocks” that are attached to every function. These logical clocks may be related to physical clocks which correspond to periods of functions. These periods are defined by automatic control engineers and are not dependent of the platform. Such approach allows verifications on the functional specification, guaranteeing that the output events of the control system obtained “in reaction” to some input events, are consistent with the input events that triggered them. Verifying functional specifications very early in the design phase, prevents a lot of classic errors found usually later on during the implementation phase. This approach is an important step for providing “correct by construction” implementations. However, non functional specifications must also be taken into consideration. Indeed, to perform real-time schedulability analyses used to guarantee that the implementation is correct in terms of time, we need the worst case execution times (WCET) of each function and the worst case communication times (WCCT) of each dependence. Both, worst case execution and communication times are dependent of the platform. Using these worst case times, schedulability analyses are able to compute the worst case response time of each function and some end-to-end worst case execution times, possibly in the case where the implied functions are allocated to different cores. Worst case response times and end-to-end execution times must verify real-time constraints, i.e. deadlines and latencies. These constraints are imposed by automatic control engineers while they usually do not know the platform that will be used later on in the developpement process.

This is the reason why, in the non functionnal specifications we need precise models that encompass important features found at different levels of the platform architecture, e.g. at a high level the number of cores and communications mediums, at a low level the structure of the caches, pipelines, etc. Depending on the complexity of the platform the problem of estimating these worst case times may be more or less difficult. In the case of simple predictable processors and buses, both used presently in the industry for critical avionics and railways applications, the estimation of worst case times is relatively easy. For this purpose we use, for example, static analyses or techniques based on measurements for estimating WCETs. However, due to the ever increasing smartphone market, the microprocessor industry provides more and more general purpose platforms based on multicore and, in a near future, based on manycore. These platform have complex architectures that are not predictable due to, e.g. multiple levels of cache and pipeline, speculative branching, communicating through shared memory or/and through a network on chip, etc. Therefore, nowadays the CPS industry has to face the great challenge of using such off the shelf platforms and consequently to estimate the corresponding worst case times of the programs (functions) that they will execute.

From functional and non functional specifications of the design phase we intend to synthesize, as automatically as possible, based on the real-time schedulability theory, an implementation that is correct by construction. This synthesizing process is close to the process used in language compilation but, in addition, it must take into account more complex non functional specifications. On the other hand, when platforms are not predictable an alternative to the classic estimation of worst case times mentioned previously, consists in reformulating the different problems in a probabilistic framework.

The overall objectives given above lead to three main research programs that are detailed below.

# 3. Research Program

## 3.1. The Algorithm-Architecture Adequation methodology and Real-Time Scheduling

**Participants:** Liliana Cucu, Dumitru Potop Butucaru, Yves Sorel.

The Algorithm-Architecture Adequation (AAA) methodology relies on distributed real-time schedulability and optimization theories to map efficiently an algorithm model to an architecture model.

The algorithm model which describes the functional specifications of the applications, is an extension of the well known data-flow model from Dennis [14]. It is a directed acyclic hyper-graph (DAG) that we call "conditioned factorized data dependence graph", whose vertices are functions and hyper-edges are directed "data or control dependences" between functions. The data dependences define a partial order on the functions execution. The basic data-flow model was extended in three directions: first infinite (resp. finite) repetition of a sub-graph pattern in order to specify the reactive aspect of real-time systems (resp. in order to specify the finite repetition of a sub-graph consuming different data similar to a loop in imperative languages), second "state" when data dependences are necessary between different infinite repetitions of the sub-graph pattern introducing cycles which must be avoided by introducing specific vertices called "delays" (similar to z -n in automatic control), third "conditioning" of a function by a control dependence similar to conditional control structure in imperative languages, allowing the execution of alternative subgraphs. Delays combined with conditioning allow the programmer to specify automata used for describing "mode changes".

The architecture model which describes the non functional specifications is, in the simplest case, a directed graph whose vertices are of two types: "processor" (one sequencer of functions, several sequencers of communications and distributed or shared memories) and "medium" (multiplexers and demultiplexers), and whose edges are directed connections. With such model it is possible to describe classic heterogeneous distributed, parallel and multiprocessor platforms as well as the most recent multi/manycore platforms. The worst case times mentioned previously are estimated according to this model.

The implementation model is a graph obtained by applying an external composition law such that an architecture graph operates on an algorithm graph to give an algorithm graph while taking advantage of timing characteristics, basically periods, deadlines and WCETs. This resulting algorithm graph is built by performing spatial and timing allocations (distribution and scheduling) of algorithm graph functions on architecture graph resources, and of dependences between functions on communication media. In that context "Adequation" means to search, in the solution space of implementation graphs, one implementation graph which verifies real-time constraints and, in addition, minimizes some criteria. These criteria consists in the total execution time of the algorithm executed on the architecture, the number of computing or communication resources, etc. Below, we describe distributed real-time schedulability analyses and optimization techniques suited for that purposes.

We address two main issues: uniprocessor and multiprocessor real-time scheduling for which some real-time constraints are of high criticality, i.e. they must be satisfied otherwise dramatic consequences occur.

In the case of uniprocessor real-time scheduling, besides the usual deadline constraint, often equal to the period of each task, i.e. a function with timing characteristics, we take into consideration dependences beetween tasks, and possibly several latencies. The latter are “end-to-end” constraints that may have complex relationships. Dealing with multiple real-time constraints raises the complexity of the scheduling problems. Moreover, costs of the Real-Time Operating System (RTOS) and of preemptions lead to, at least, a waste of resources due to their approximation in the WCET (Worst Execution Time) of each task, as proposed by Liu and Layland in their seminal article [21]. This is the reason why we first studied non-preemptive real-time scheduling with dependences, periodicities, and latencies constraints. Although a bad approximation of costs of the RTOS and of preemptions, may have dramatic consequences on real-time scheduling, there are only few researches on this topic. Thus, we investigated preemptive real-time scheduling while taking into account its cost which is very difficult to determine because it varies according to the instance (job) of each task. This latter is integrated in the schedulability conditions, and in the corresponding scheduling algorithms we propose. More generally, we integrate in schedulability analyses costs of the RTOS and of preemptions.

In the case of multiprocessor real-time scheduling, we chose to study first the “partitioned approach”, rather than the “global approach”, since the latter uses task migrations whose cost is prohibitive for current commercial processors, even for the more recent many/multicore. The partitioned approach enables us to reuse the results obtained in the uniprocessor case in order to derive solutions for the multiprocessor case. We consider also the semi-partitioned approach which allows only some migrations in order to minimize their costs. In addition, to satisfy the multiple real-time constraints mentioned in the uniprocessor case, we have to minimize the total execution time (makespan) since we deal with automatic control applications involving feedback loops. The complexity of such minimization problem increases because the cost of interprocessor communications (through buses in a multi-processor or routers in a manycore) must be taken into account. Furthermore, the domain of embedded systems leads to solving minimization resources problems. Since both optimization problems are NP-hard we develop exact algorithms (ILP, B & B, B & C) which are optimal for simple problems, and heuristics which are sub-optimal for realistic problems corresponding to industrial needs. Long time ago we proposed a very fast “greedy” heuristics whose results were regularly improved, and extended with local neighborhood heuristics, or used as initial solutions for metaheuristics.

Besides the spatial dimension (distributed) of the real-time scheduling problem, other important dimensions are the type of communication mechanisms (shared memory vs. message passing), or the source of control and synchronization (event-driven vs. time-triggered). We explore real-time scheduling on architectures corresponding to all combinations of the above dimensions. This is of particular impact in application domains such as railways and avionics.

## 3.2. Probabilistic Worst Case Reasoning for Real-Time Systems

**Participants:** Liliana Cucu, Robert Davis, Yves Sorel.

The arrival of modern hardware responding to the increasing demand for new functionalities exacerbates the limitations of the current worst-case real-time reasoning, mainly to the rarity of worst-case scenarios. Several solutions exist to overcome this important pessimism and our solution takes into account the extremely low probability of appearance of a worst-case scenario within one hour of functioning ($10^{-45}$), compared to the certification requirements for instance ($10^{-9}$ for the highest level of certification in avionics). Thus we model and analyze real-time systems with time parameters described by using probabilistic models. Our results for such models address both schedulability analyses as well as timing analyses. Both such analyses are impacted by existing misunderstanding. The independence between tasks is a property of real-time systems that is often used for its basic results. Any complex model takes into account different dependences caused by sharing resources other than the processor. On another hand, the probabilistic operations require, generally, the (probabilistic) independence between the random variables describing some parameters of a probabilistic real-time system. The main (original) criticism to probabilistic is based on this hypothesis of independence judged too restrictive to model real-time systems. In reality the two notions of independence are different. Providing arguments to underline this confusion is at the center of our dissemination effort in the last years.

We provide below the bases driving our current research as follows:

- *Optimality of scheduling algorithms* stays an important aspect of the probabilistic real-time systems, especially that the introduction of probabilistic time parameters has a direct impact on the optimality of the existing scheduling algorithms. For instance Rate Monotonic scheduling policy is no longer optimal in the case of one processor when a preemptive fixed-priority solution exists. We expect other classes of algorithms to lose their optimality and we concentrate our efforts to propose new scheduling solutions in this context [22].
- *Increased complexity of schedulability analysis* due to the introduction of probabilistic parameters requires appropriate complexity reasoning, especially with the emergence of probabilistic schedulability analyses for mixed-criticality real-time systems [23]. Moreover the real-time applications are rarely independent and precedence constraint using graph-based models are appropriate in this context. Precedence constraints do decrease the number of possible schedulers, but they also imposes an "heritage" of probabilistic description from execution times to release times for instance.
- *Proving feasibility intervals* is crucial for these approaches that are often used in industry on top of simulation. As worst-case situations are rare events, then observing them or at least observe those events that do provoke later the appearance of worst-case situations is difficult. By proposing an iterative process of composition between different statistical models [17], we provide the basis to build a solution to this essential problem to prove any probabilistic real-time reasoning based on measurements.
- *Providing representativeness* of a measurement-based estimator is the final proof that a probabilistic worst-case reasoning may receive. Our first negative results [24] indicate that the measurement protocol is tighly connected to the statistical estimator and that both must verified properties of reproducibility in order to contribute to a convergence proof.

## 3.3. Real-Time Systems Compilation

**Participant:** Dumitru Potop Butucaru.

In the early days of embedded computing, most software development activities were manual. This is no longer true at the low level, where manual assembly coding has been almost completely replaced with the combined use of so-called “high-level” languages (C, Ada, *etc.*) and the use of compilers. This was made possible by the early adoption of standard interfaces that allowed the definition of economically-viable compilation tools with a large-enough user base. These interfaces include not only the programming languages (C, Ada, *etc.*), but also relatively stable microprocessor instruction set architectures (ISAs) or executable code formats like ELF.

The paradigm shift towards fully automated code generation is far from being completed at the system level, mainly due to the slower introduction of standard interfaces. This also explains why real-time scheduling has historically dedicated much of its research effort to verifying the correctness of very abstract and relatively standard implementation models (the task models). The actual construction of the implementations and the abstraction of these implementations as task models drew comparatively less interest, because they were application-dependent and non-portable.

But today the situation is bound to change. First, automation can no longer be avoided, as the complexity of systems steadily increases in both specification size (number of tasks, processors, etc.) and complexity of the objects involved (parallelized dependent tasks, multiple modes and criticalities, many-cores, *etc.*). Second, fully automated implementation is attainable for industrially significant classes of systems, due to significant advances in the standardization of both specification languages (Simulink, Scade, etc.) and of implementation platforms (ARINC, AUTOSAR, *etc.*).

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 – including the AAA methodology – 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. Similarly, 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/platform 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. In particular, resource allocation, timing analysis, and code generation must be tightly integrated to ensure that generated code (including communication and synchronization primitive calls) satisfies the timing hypotheses used for scheduling. More generally, and unlike in classical compilers, the allocation and scheduling algorithms must take into account a variety of non-functional requirements, such as real-time constraints, criticality/partitioning, preemptability, allocation constraints, *etc.* As the accent is put on the respect of requirements (as opposed to optimization of a metric, like in classical compilation), resulting scheduling problems are quite different from those of classical compilation.

We have designed and built a prototype real-time systems compiler, called LoPhT, for statically scheduled real-time systems. Results on industrial case studies are encouraging, hinting not only at the engineering potential of the approach, but also at the scientific research directions it opens.

One key issue here is sound hardware/platform abstraction. To prove that it is possible to reconcile performance with predictability in a fully automatic way, we started in the best possible configuration with industrial relevance: statically-scheduled software running on very predictable (yet realistic) platforms. Already, in this case, platform modeling is more complex than the one of classical compilation [1] or real-time scheduling. [2] The objective is now to move beyond this application class to more dynamic classes of specifications implementations, but without losing too much of the predictability and/or effciency.

Efficiency is also a critical issue in practical systems design, and we must invest more in the design of optimizations that improve the *worst-case* behavior of applications and take into account non-functional requirements in a *multi-objective optimization* perspective, but while remaining in the class of low-complexity heuristics to ensure scalability. Optimizations of classical compilation, such as loop unrolling, retiming, and inlining, can serve as inspiration.

Ensuring the safety and efficiency of the generated code cannot be done by a single team. Collaborations on the subject will have to cover at least the following subjects: the interaction between real-time scheduling

[1] Because safe timing accounting is needed.

[2] The compiler must perform safe timing accounting, and not rely on experience-derived margins.

and WCET analysis, the design of predictable hardware and software architectures, programming language support for efficient compilation, and formally proving the correctness of the compiler.

# 4. Application Domains

## 4.1. Avionics

**Participants:** Liliana Cucu, Keryan Didier, Adriana Gogonel, Cristian Maxim, Dumitru Potop Butucaru, Yves Sorel.

A large number of our activities, in analysis, modelling, design and implementation of real-time embedded systems addresses specific applications mainly in the avionics field (with partners such as Airbus, Thales, Safran, etc.) (in the ASSUME project 9.2.1.1).

## 4.2. Many-Core Embedded Architectures

**Participants:** Liliana Cucu, Keryan Didier, Dumitru Potop Butucaru, Yves Sorel.

The AAA approach (fitting embedded applications onto embedded architectures) requires a sufficiently precise description of (a model of) the architecture (description platform). Such platforms become increasingly heterogeneous, and we had to consider a number of emerging ones with that goal in mind, such as Kalray MPPA (in the ASSUME project 9.2.1.1).

## 4.3. Railways

**Participants:** Liliana Cucu, Adriana Gogonel, Walid Talaboulma.

The statistical estimation of bounds on the execution time of a program on a processor is applied in the context of railroad crossing in the context of the collaborative project DEPARTS 9.1.2.2.

# 5. Highlights of the Year

## 5.1. Highlights of the Year

This is the last activity report of the team AOSTE2 since it ends in 2018.

The ATT StatInf project, prepared by Liliana Cucu-Grosjean and Adriana Gogonel has been accepted in July 2018. The associated start-up creation has been selected for participation to the Digital Start-up program (jointly supported by EMLyon and Inria). The start-up will be created beginning of 2019 by Adriana Gogonel, Cristian Maxim and Liliana Cucu-Grosjean as founding members.

# 6. New Software and Platforms

## 6.1. SynDEx

KEYWORDS: Distributed - Optimization - Real time - Embedded systems - Scheduling analyses
SCIENTIFIC DESCRIPTION: SynDEx is a system level CAD software implementing the AAA methodology for rapid prototyping and for optimizing distributed real-time embedded applications. It is developed in OCaML.

Architectures are represented as graphical block diagrams composed of programmable (processors) and non-programmable (ASIC, FPGA) computing components, interconnected by communication media (shared memories, links and busses for message passing). In order to deal with heterogeneous architectures it may feature several components of the same kind but with different characteristics.

Two types of non-functional properties can be specified for each task of the algorithm graph. First, a period that does not depend on the hardware architecture. Second, real-time features that depend on the different types of hardware components, ranging amongst execution and data transfer time, memory, etc.. Requirements are generally constraints on deadline equal to period, latency between any pair of tasks in the algorithm graph, dependence between tasks, etc.

Exploration of alternative allocations of the algorithm onto the architecture may be performed manually and/or automatically. The latter is achieved by performing real-time multiprocessor schedulability analyses and optimization heuristics based on the minimization of temporal or resource criteria. For example while satisfying deadline and latency constraints they can minimize the total execution time (makespan) of the application onto the given architecture, as well as the amount of memory. The results of each exploration is visualized as timing diagrams simulating the distributed real-time implementation.

Finally, real-time distributed embedded code can be automatically generated for dedicated distributed real-time executives, possibly calling services of resident real-time operating systems such as Linux/RTAI or Osek for instance. These executives are deadlock-free, based on off-line scheduling policies. Dedicated executives induce minimal overhead, and are built from processor-dependent executive kernels. To this date, executives kernels are provided for: TMS320C40, PIC18F2680, i80386, MC68332, MPC555, i80C196 and Unix/Linux workstations. Executive kernels for other processors can be achieved at reasonable cost following these examples as patterns.
FUNCTIONAL DESCRIPTION: Software for optimising the implementation of embedded distributed real-time applications and generating efficient and correct by construction code
NEWS OF THE YEAR: We improved the distribution and scheduling heuristics to take into account the needs of co-simulation.

- Participant: Yves Sorel
- Contact: Yves Sorel
- URL: http://www.syndex.org

## 6.2. EVT Kopernic

KEYWORDS: Embedded systems - Worst Case Execution Time - Real-time application - Statistics
SCIENTIFIC DESCRIPTION: The EVT-Kopernic tool is an implementation of the Extreme Value Theory (EVT) for the problem of the statistical estimation of worst-case bounds for the execution time of a program on a processor. Our implementation uses the two versions of EVT - GEV and GPD - to propose two independent methods of estimation. Their results are compared and only results that are sufficiently close allow to validate an estimation. Our tool is proved predictable by its unique choice of block (GEV) and threshold (GPD) while proposant reproducible estimations.
FUNCTIONAL DESCRIPTION: EVT-Kopernic is tool proposing a statistical estimation for bounds on worst-case execution time of a program on a processor. The estimator takes into account dependences between execution times by learning from the history of execution, while dealing also with cases of small variability of the execution times.
NEWS OF THE YEAR: Any statistical estimator should come with an representative measurement protocole based on the processus of composition, proved correct. We propose the first such principle of composition while using a Bayesien modeling taking into account iteratively different measurement models. The composition model has been described in a patent submitted this year with a scientific publication under preparation.

- Participants: Adriana Gogonel and Liliana Cucu
- Contact: Adriana Gogonel
- URL: http://inria-rscript.serveftp.com/

## 6.3. LoPhT-manycore

*Logical to Physical Time compiler for many cores*

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

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

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

The last point is characteristic to Lopht-manycore. Existing methods for schedulability analysis and real-time software synthesis assume the existence of a high-level timing characterization that hides much of the hardware complexity. For instance, a common hypothesis is that synchronization and interference costs are accounted for in the duration of computations. However, the high-level timing characterization is seldom (if ever) soundly derived from the properties of the platform and the program. In practice, large margins (e.g. 100%) with little formal justification are added to computation durations to account for hidden hardware complexity. Lopht-manycore overcomes this limitation. Starting from the worst-case execution time (WCET) estimations of computation operations and from a precise and safe timing model of the platform, it maintains a precise timing accounting throughout the mapping process. To do this, timing accounting must take into account all details of allocation, scheduling, and code generation, which in turn must satisfy specific hypotheses.
FUNCTIONAL DESCRIPTION: Accepted input languages for functional specifications include dialects of Lustre such as Heptagon and Scade v4. To ensure the respect of real-time requirements, Lopht-manycore pilots the use of the worst-case execution time (WCET) analysis tool (ait from AbsInt). By doing this, and by using a precise timing model for the platform, Lopht-manycore eliminates the need to adjust the WCET values through the addition of margins to the WCET values that are usually both large and without formal safety guarantees. The output of Lopht-manycore is formed of all the multi-threaded C code and configuration information needed to allow compilation, linking/loading, and real-time execution on the target platform.
NEWS OF THE YEAR: In the framework of the ITEA3 ASSUME project we have extended the Lopht-manycore to allow multiple cores to access the same memory bank at the same time. To do this, the timing accounting of Lopht has been extended to take into account memory access interferences during the allocation and scheduling process. Lopht now also pilots the aiT static WCET analysis tool from AbsInt by generating the analysis scripts, thus ensuring the consistency between the hypotheses made by Lopht and the way timing analysis is performed by aiT. As a result, we are now able to synthesize code for the computing clusters of the Kalray MPPA256 platform. Lopht-manycore is evaluated on avionics case studies in the perspective of increasing its technology readiness level for this application class.

- Participants: Dumitru Potop-Butucaru and Keryan Didier
- Contact: Dumitru Potop-Butucaru

# 7. New Results

## 7.1. Uniprocessor Mixed-Criticality Real-Time Scheduling

**Participants:** Liliana Cucu, Robert Davis, Mehdi Mezouak, Yves Sorel.

In the context of the FUI CEOS project 9.1.1.1, last year we tranformed the PX4 autopilot free software program in a graph of tasks. In this project our main goal is to perform a real-time schedulability analysis on this program in order to prove that the autopilot will meet all its deadlines when it will operate in the multirotor drone the project is intended to built. The tasks will be executed on a Pixhawk electronic board based on an ARM Cortex M4 microprocessor running on the NuttX OS.

We start by determinating the period and measuring the average execution time of each task which is less than the worst case execution time (WCET). Then, using these periods and these measured execution times we perform an online schedulability analysis using a rate monotonic policy (RM) that shown the set of tasks is not schedulable. Consequently, we informed the partners of the CEOS project that the present version of PX4 is not real-time.

Presently, we are transforming the original set of tasks into a set of real-time tasks. To achieve this goal, we associate to every task a periodic high resolution timer corresponding to the period of the task. Each timer generates an interruption when it expires and the task is put in the ready task queue. The scheduler of NuttX will choose in this queue the task to be executed. In order to validate this transformation we operated the multirotor drone in a simulation tool composed of Gazebo for the geometrical environment of the drone and of the Ground Control Station for setting and controlling the drone. We performed two kinds of simulations, a software in the loop simulation (SitL) which simulates the Pixhawk board, the sensors and the actuators, and a hardware in the loop simulation (HitL) which simulates only the sensors and the actuators, whereas the PX4 program runs on the Pixhawk board. We tested the set of real-time tasks in SitL and we are presently testing them in HitL.

Since we can easily change the period of every task, we plan to modify the periods to make the set of real-time tasks schedulable using an online RM schedulability analysis.

In order to manage high criticality real-time tasks we plan to use an offline scheduler whose scheduling table is generated by an offline schedulability analysis tool that is developped in the team. We plan to modify NuttX in order to support such scheduler.

Finally, in order to complete the real-time schedulability analysis of PX4, we estimate the worst case execution time (WCET) of each task. This problem is complex due to the multiple possible paths in a task as well as the different data it consumes. Moreover, the processor and/or the microcontroller itself may have some features like memory contentions, bus accesses, caches, pipelines, speculative branchings that increase the difficulty to determine WCETs. All these variabilities lead us to introduce statistical reasoning in characterizing the timing behavior (WCET, schedulability analyses) of mixed-criticality real-time applications. The isolated execution times of the programs have indicated large variations indicating expected larger variability in real execution scenarios. In order to decrease the pessimism of the statistical bounds, we are adapting our models to move towards multi-variate approaches.

## 7.2. Multiprocessor Real-Time Scheduling

**Participants:** Slim Ben Amor, Evariste Ntaryamira, Salah Eddine Saidi, Yves Sorel, Walid Talaboulma.

The last part of the PhD thesis of Salah Eddine Saidi, was dedicated to the parallelization of FMI-based co-simulation under real-time constraints. More precisely we address HiL (Hardware in the Loop) co-simulation where a part of the co-simulation is replaced by its real counterpart which is physically available. The real and simulated parts have to exchange data during the execution of the co-simulation under real-time constraints. In other words, the inputs (resp. ouputs) of the real part are sampled periodically, sending (resp. receiving) data to (resp. from) the simulated part. Every periodic data exchange defines a set of real-time constraints to be satisfied by the simulated part. We proposed a method for defining these real-time constraints and propagating them to all the data dependent functions that specify the co-simulation (simulated part). Starting from these constraints we have to schedule the FMI-based co-simulation on a multi-core. We propose an ILP-based algorithm as well as a heuristic that allow the execution of the co-simulation on a multi-core processor while ensuring the previously defined real-time constraints are respected [6]. The proposed heuristic is a list scheduling heuristic. It builds the multi-core schedule iteratively. At each iteration, a list of candidate functions is constructed. The heuristic computes the priority for each candidate function on every core and selects the core for the which the priority is maximized. The priority of a function is a dynamic priority as its computation depends on the partial scheduling solution that has already been computed.

All works achieved by Salah Eddine Saidi on the parallelization of FMI-based co-simulation of numerical models were presented in his PhD thesis defense and manuscript [1].

Avionics applications are based on the specification of “data chains”. Every data chain is a sequence of periodic real-time communicating tasks that are processing the data from sensors up to actuators. Such data chain determines an order in which the tasks propagate data but not in which they are executed. Indeed, inter-task communication and scheduling are independent. We focus on the latency computation, considered as the time elapsed from getting the data from an input and processing it to an output of a data chain. We propose a method for the worst-case latency computation of data chains composed of periodic tasks and executed by a partitioned fixed-priority preemptive scheduler upon a multiprocessor platform [5].

The PhD thesis of Slim Ben Amor is dedicated to the study of multiprocessor scheduling of real-time systems in presence of precedence constraints. This year we have proposed new models [10] for dependent real-time task with probabilistic worst-case execution time (WCET) that are scheduled using a partitioned reasoning. We explore existing solutions from [15] as the closest problem to our dependent task scheduling on multiprocessor and we study their extension to probabilistic models. We conclude that the probabilistic extension would be very difficult with heavy computation since the deterministic solution is based on the resolution of complex ILP optimization problem. Then, we decide to build a new solution to the deterministic problem that should be simple to extend to probabilistic problem. The proposed solution [11] consists of calculating the response time of each sub-tasks in a given DAG task taking in consideration preemptions caused by higher priority sub-tasks executed on the same processor. Then, we evaluate the global response time of the whole graph layer by layer, which allows deciding the schedulability of the entire system.

During the third year of Walid Talaboulma PhD thesis, we continued exploring solutions to make the WCET (Worst Case Execution Time) estimation as independent as possible with respect to the memory accesses. WCET analysis done on a unicore processor (in isolation) is not sufficient when we run our tasks on a multicore processors, the problem of Co-runner interference arises due to contention in shared hardware. Our solution is based on the generation of programs memory access profile, that we obtain by running tasks on a cycle accurate System Simulator, with a precise cycle accurate model of DDRAM memory controller and a full model of memory hierarchy including caches and main memory devices, and we log every memory event that occurs inside the simulation. Our solution does not necessarily require modifications of software layer, or recompilation of task code. We use those profiles to account for co runners interference and add it to WCET value obtained in isolation, and by updating our schedule, we can also insert idle times at correct scheduling events to decrease the interference.

The PhD thesis of Evariste Ntaryamira is dedicated to the study of multiprocessor real-time systems while ensuring the data freshness. This year we have underlined the difficulty of this scheduling problem [13], [8] while proposing a model to include both time and data constraints. We explore existing solutions from [16] as the closest problem to our data-dependent scheduling problem. The case study associated to this thesis is jointly prepared with the members of the RITS Inria team.

## 7.3. Safe Parallelization of Hard Real-Time Avionics Software

**Participants:** Keryan Didier, Dumitru Potop Butucaru.

This work took place in the framework of the ITEA3 ASSUME project, which funds the PhD thesis of Keryan Didier, and in close collaboration with Inria PARKAS, Airbus, Safran Aircraft Engines, and Kalray.

The key difficulty of real-time scheduling is that timing analysis and resource allocation depend on each other. An exhaustive search for the optimal solution not being possible for complexity reasons, heuristic approaches are used to break this dependency cycle. Two such approaches are typical in real-time systems design. The first approach uses unsafe timing characterizations for the tasks (e.g., measurements) to build the system, and then checks the respect of real-time requirements through a global timing analysis. The second approach uses a formal model of the hardware platform enabling timing characterizations that are safe for all possible resource allocations (worst-case bounds).

So far, the practicality of the second approach had never been established. Automated real-time parallelization flows still relied on simplified hypotheses ignoring much of the timing behavior of concurrent tasks, communication and synchronization code. And even with such unsafe hypotheses, few studies and tools considered the—harmonic—multiperiodic task graphs of real-world control applications, and the problem of statically managing all their computational, memory, synchronization and communication resources.

This year, we presented the first demonstration of the feasibility of the second approach, showing good practical results for classes of real-world applications and multiprocessor execution platforms whose timing predictability allows keeping pessimism under control. This requires something that is missing in previous work: *the tight orchestration of* **all** *implementation phases*: WCET analysis, resource allocation, generation of *glue code* ensuring the sequencing of tasks on cores and the synchronization and memory coherency between the cores, compilation and linking of the resulting C code. This orchestration is conducted on very detailed timing model that considers both the tasks and the generated glue code, and which includes resource access interferences due to multi-core execution. While orchestration is our main contribution, it should not be understood as a mere combination of existing tools and algorithms. The whole point of our approach is to carefully coordinate every analysis, mapping and code generation phase to enable predictable execution and to keep pessimism under control. To this end, we contributed application normalization phase to facilitate timing analysis, an original code generation algorithm designed to provide mapping-independent worst-case execution time bounds, and new real-time scheduling algorithms capable of orchestrating memory allocation and scheduling.

Our flow scales to an avionics application comprising more than 5000 unique nodes, targeting the Kalray MPPA 256 many-core platform, selected for its timing predictability. First results are presented in the report [9].

## 7.4. Real-time Platform Modeling

**Participants:** Fatma Jebali, Dumitru Potop Butucaru.

This work took place in the framework of the ITEA3 ASSUME project, which funds the post-doc of Fatma Jebali.

One key difficulty in embedded systems design is the existence of multiple models of the same hardware system, developed separately, at different abstraction levels, and used in various phases of the design flow. In the design of real-time embedded systems, we can identify, among other:

- Cycle-accurate system models used to perform fine-grain hardware simulation, mostly during HW and driver design phases. These models provide an exact functional and temporal representation of system execution.
- Microarchitectural models used for pipeline simulation during WCET (*Worst-Case Execution Time*) analysis [19], [20], [18]. These models are used to compute safe over-approximations of the duration of a sequential piece of code, i.e., one function running without interruption on a processor core). To provide precise results, these models preserve much of the microarchitectural detail of processor pipelines and memory hierarchy (e.g. cache states, data transfer latencies).

Both simulation models usually have cyclic activation patterns, but establishing semantic consistency between them is challenging for several reasons. First, the activation pattern, which is the logical time base of the simulation, depends on the abstraction level. In cycle-accurate models, simulation cycles correspond to hardware clock ticks, whereas in WCET analysis models they correspond to changes in the program counter of the sequential program. Second, data abstractions are different in the two simulation models. Cycle-accurate simulators are often also *bit-accurate*, *i.e.* provide exactly the same results as the actual hardware. By comparison, pipeline simulators in WCET analysis abstract away most data types and related operators, typically retaining only Booleans, which can be exploited at analysis time. Last, but not least, the simulators are usually pieces of C/C++ code manually written by different teams or obtained through complex translation processes from high-level Architecture Description Languages (ADLs) that may not have a clear semantics. Formally relating such pieces of code is difficult.

This year we proposed a method to ensure the semantic consistency between the two HW models we consider, focusing on time abstraction issues. Our method relies on *desynchronization* theory [25], which defines sufficient properties ensuring that a synchronous model can be seen as an asynchronous Kahn Process Network (KPN). When a synchronous HW model satisfies these properties, any scheduling of its computations that is compatible with data dependencies will produce the same result (a property known as scheduling-independence). We showed how to control scheduling through changes of the logical time base of the model prior to code generation using a synchronous language compiler. In particular, a careful choice of the logical time base allows us to produce, from the same model, either a cycle-accurate simulator, or the one needed for WCET analysis. In conjunction with some data abstraction, this logical time manipulation allows the synthesis of semantically consistent simulators from a single model.

Furthermore, we can ensure by construction that synchronous models satisfy the properties required by desynchronization theory. To this end, we introduced a new hardware modelling language, named xMAStime, allowing the compositional modeling of systems satisfying the required properties. Results were presented at the ACSD'18 conference [4].

# 8. Bilateral Contracts and Grants with Industry

## 8.1. Bilateral Contracts with Industry

The IFPEN grant which started on December 2014 and ended on February 2018, provides full support for the PhD thesis of Salah Eddine Saidi. The thesis concerns the automatic parallelization and scheduling approaches for co-simulation of numerical models on multi-core processors. The goal of the first research topic is to propose multi-core scheduling solutions for the co-simulation in order to accelerate its execution. The second research topic aims at proposing multi-core scheduling solutions in order to enable the execution of co-simulation under real-time constraints in the context of Hardware-in-the-Loop validation.

# 9. Partnerships and Cooperations

## 9.1. National Initiatives

### *9.1.1. FUI*

#### *9.1.1.1. CEOS*

**Participants:** Slim Ben Amor, Liliana Cucu, Cristian Maxim, Mehdi Mezouak, Yves Sorel, Walid Talaboulma.

This project was started on May 2017. Partners of the project are: ADCIS, ALERION, Aéroports de Lyon, EDF, ENEDIS, RTaW, EDF, Thales Communications and Security, ESIEE engineering school and Lorraine University. The CEOS project delivers a reliable and secure system of inspections of pieces of works using professional mini-drone for Operators of Vital Importance coupled with their Geographical Information System. These inspections are carried out automatically at a lower cost than current solutions employing helicopters or off-road vehicles. Several software applications proposed by the industrial partners, are developed and integrated in the drone, within an innovative mixed-criticality approach using multi-core platforms.

##### *9.1.1.2. WARUNA*

**Participants:** Liliana Cucu, Adriana Gogonel, Yves Sorel, Walid Talaboulma.

This FUI funded project was started on September 2015 and it is preparing its final conclusions for the beginning of 2019. It has targeted the creation of the framework Time4Sys within the PolarSys project [12]. This open source framework allows timing analyses from models to simulation for different application domains like avionics, railways, medical, aerospace, automotive, etc. and it is available at https://www.polarsys.org/time4sys.

### *9.1.2. PIA*

##### *9.1.2.1. ES3CAP*

**Participants:** Keryan Didier, Dumitru Potop Butucaru.

The objectives of the ES3CAP (Embedded Smart Safe Secure Computing Autonomous Platform) project are to:

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

##### *9.1.2.2. DEPARTS*

**Participants:** Liliana Cucu, Adriana Gogonel, Walid Talaboulma.

This BGLE funded project of the national support programme Investissements d'Avenir has started on October 1st, 2012 and provided its final conclusions on December 2018. Inria has provided a final prototype version of the EVT Kopernic tool taking into account homogenous variation factors for the execution times. Swapping algorithms allowing WCET decrease are currently finalized within the PhD thesis of Walid Talaboulma with a defense expected during the spring of 2019.

## 9.2. European Initiatives

### *9.2.1. Collaborations in European Programs, Except FP7 & H2020*

##### *9.2.1.1. ASSUME*

**Participants:** Keryan Didier, Fatma Jebali, Dumitru Potop Butucaru.

Program: ITEA

Project acronym: ASSUME

Project title: Affordable Safe and Secure Mobility Evolution

Duration: September 2015 - August 2018

Coordinator: Daimler

Other partners: among 38 partners Absint, Ansys, Airbus, Kalray, Safran, Thales, ENS, KTH, FZI, etc.

Abstract: Future mobility solutions will increasingly rely on smart components that continuously monitor the environment and assume more and more responsibility for a convenient, safe and reliable operation. Currently the single most important roadblock for this market is the ability to come up with an affordable, safe multi-core development methodology that allows industry to deliver trustworthy new functions at competitive prices. ASSUME will provide a seamless engineering methodology, which addresses this roadblock on the constructive and analytic side.

### *9.2.2. Collaborations with Major European Organizations*

University of York: Real-Time System Group (UK)

Uncertainties in real-time systems: the utilization of extreme value theory has received increased efforts from our community and more rigorous principles are needed for its full understanding. Our two research teams have gathered these principles in several publications.

# 10. Dissemination

## 10.1. Promoting Scientific Activities

### *10.1.1. Scientific Events Organisation*

#### *10.1.1.1. General Chair, Scientific Chair*

Liliana Cucu-Grosjen is member of the steering committees of the following conferences and workshops: RTSS, RTAS, RTNS, WMC, RTSOPS.

Rob Davis is member of the steering committees of the following conferences and workshops: RTSS, RTAS, RTNS, WMC, RTSOPS.

### *10.1.2. Scientific Events Selection*

#### *10.1.2.1. Chair of Conference Program Committees*

Liliana Cucu-Grosjean has served as PC co-chair for the 14th edition of WFCS 2018 in Imperia, Italy.

Adriana Gogonel has served as PC co-chair for the 12th Junior Researcher Workshop on Real-Time Computing (JWRTC) edition of in Poitiers, France.

#### *10.1.2.2. Member of the Conference Program Committees*

Liliana Cucu: RTAS, RTNS, WFCS.

Robert Davis: RTSS, RTAS, RTNS.

Adriana Gogonel: ACM RACS, WMC, JWRTC.

Dumitru Potop Butucaru: ACSD, EMSOFT.

Yves Sorel: DASIP.

#### *10.1.2.3. Reviewer*

All members of the team are regularly serving as reviewers for the main scientific events of our domain: RTSS, RTAS, RTCSA, RTNS, DATE, ETFA, EMSOFT, DASIP, etc.

### *10.1.3. Journal*

#### *10.1.3.1. Member of the Editorial Boards*

Liliana Cucu-Grosjean has served as guest editor for the Journal of Real-Time Systems

#### *10.1.3.2. Reviewer - Reviewing Activities*

All members of the team are regularly serving as reviewers for the main journals of our domain: Information Processing Letter, Journal of Heuristics, Journal of Real-Time Systems, Journal of Systems Architecture, Journal of Signal Processing Systems, Leibniz Transactions on Embedded Systems, IEEE Transactions on Industrial Informatics, IEEE Transactions on Computers, Theoretical Computer Science.

### *10.1.4. Scientific Expertise*

Yves Sorel: Steering Committee of System Design and Development Tools Group of Systematic Paris-Region Cluster.

Yves Sorel: Steering Committee of Technologies and Tools Program of SystemX Institute for Technological Research (IRT).

### *10.1.5. Research Administration*

Liliana Cucu-Grosjean is member of Inria Evaluation Commission, co-chair of Inria Committee on gender equality and equal oportunities, and member of the CLHCST.

Dumitru Potop Butucaru is member of mobility grant commission for postdocs and invited professors.

## 10.2. Teaching - Supervision - Juries

### *10.2.1. Teaching*

Master: Slim Ben Amor, Machine learning (practice sessions), 20H, M1, ESIEE Engineering School, Noisy-Le-Grand, France.

Master: Liliana Cucu, Distributed Databases and Statistics in Computer Science, 64h, U. Dunarea de Jos, Romania (Invited Professor).

Master: Liliana Cucu-Grosjean, Graph Theory, 32H, M1, ESIEE Engineering School, Cergy Pontoise, France.

Master: Adriana Gogonel, Machine learning, 32H, M1, ESIEE Engineering School, Noisy le Grand, France.

Master: Dumitru Potop Butucaru, A synchronous approach to the design of embedded real-time systems, 30h, M1, EPITA Engineering School, Paris France.

Master: Cristian Maxim, Graph Theory, 12H, M1, ESIEE Engineering School, Cergy Pontoise, France.

Master: Yves Sorel, Optimization of distributed real-time embedded systems, 38H, M2, University of Paris Sud, France.

Master: Yves Sorel, Safe design of reactive systems, 18H, M2, ESIEE Engineering School, Noisy-Le-Grand, France.

### *10.2.2. Supervision*

PhD: Salah-Edinne Saidi, Distributed real-time scheduling for the co-simulation of multiple control models, UPMC, defended April 2018, co-supervised by Nicolas Pernet (IFPEN) and Yves Sorel.

PhD in progress: Slim Ben Amor, Schedulability analysis of probabilistic real-time tasks under end to end constraints, UPMC, started on September 2016, supervised by Liliana Cucu.

PhD in progress: Keryan Didier, Formal certification of real-time implementations, UPMC, started November 2015, supervised by Dumitru Potop Butucaru.

PhD in progress: Evariste Ntaryamira, Analysis of embedded systems with time and security constraints, UPMC, started on January 2017, supervised by Liliana Cucu and Rachel Akimana.

PhD in progress: Walid Talaboulma, Probabilistic timing analysis in presence of dependences, UPMC, started November 2015, co-supervised by Liliana Cucu and Adriana Gogonel.

PhD in progress: Salah-Edinne Saidi, Distributed real-time scheduling for the co-simulation of multiple control models, UMPC, started December 2014, co-supervised by Nicolas Pernet (IFPEN) and Yves Sorel.

#### *10.2.3. Juries*

Liliana Cucu-Grosjean is PhD examiner for the thesis of Anh Toan Bui Long, University of Poitiers/ENSMA, defended on December 2018.

Dumitru Potop Butucaru is PhD examiner for the thesis of Jad Khatib, Université Pierre et Marie Curie/EDITE, defended September 2018. École doctorale EDITE.

Yves Sorel is Phd examiner for the thesis of Florian Greff, University of Lorraine, defended May 2018.

# 11. Bibliography

## Publications of the year

### Doctoral Dissertations and Habilitation Theses

[1] S. E. SAIDI. *Automatic Parallelization and Scheduling Approches for Co-simulation of Numerical Models on Multi-core Processors*, Université Sorbonne, April 2018, https://hal.inria.fr/tel-01895280

### Articles in International Peer-Reviewed Journals

[2] L. CUCU-GROSJEAN, N. FISHER. *Guest editorial: special issue on real time and network systems*, in "Real-Time Systems", July 2018, vol. 54, n$^{o}$ 3, pp. 605-606 [*DOI :* 10.1007/s11241-018-9309-8], https://hal.inria.fr/hal-01955995

[3] B. LESAGE, S. ALTMEYER, D. GRIFFIN, L. CUCU-GROSJEAN, R. DAVIS. *On the analysis of random replacement caches using static probabilistic timing methods for multi-path programs*, in "Real-Time Systems", April 2018, vol. 54, n$^{o}$ 2, pp. 307-388 [*DOI :* 10.1007/s11241-017-9295-2], https://hal.archives-ouvertes.fr/hal-01666091

### International Conferences with Proceedings

[4] F. JEBALI, D. POTOP-BUTUCARU. *Ensuring consistency between cycle-accurate and instruction set simulators*, in "ACSD 2018 - 18th International Conference on Application of Concurrency to System Design", Bratislava, Slovakia, June 2018, https://hal.inria.fr/hal-01959370

[5] T. KLODA, A. BERTOUT, Y. SOREL. *Latency analysis for data chains of real-time periodic tasks*, in "ETFA'2018 - IEEE International Conference on Emerging Technologies and Factory Automation", Torino, Italy, Proceedings of the 23rd IEEE International Conference on Emerging Technologies and Factory Automation, ETFA'18, September 2018, https://hal.inria.fr/hal-01939228

[6] S. E. SAIDI, N. PERNET, Y. SOREL. *Scheduling Real-time HiL Co-simulation of Cyber-Physical Systems on Multi-core Architectures*, in "RTCSA2018 - IEEE International Conference on Embedded and Real-Time Computing Systems and Applications", Hakodate, Japan, Proceedings of the 24th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications, August 2018, https://hal.inria.fr/hal-01887155

[7] J. SOUYRIS, K. DIDIER, D. POTOP-BUTUCARU, G. IOOSS, T. BOURKE, A. COHEN, M. POUZET. *Automatic Parallelization from Lustre Models in Avionics*, in "ERTS2 2018 - 9th European Congress Embedded Real-Time Software and Systems", Toulouse, France, Proceedings of the 9th European Congress on Embedded Real-Time Software and Systems (ERTS2), 3AF - Association Aéronautique Astronautique de France and SEE - Société de l'électricité, de l'électronique et des technologies de l'information et de la communication and SIA - Société de Ingénieurs de l'Automobile, January 2018, pp. 1-4, https://hal.inria.fr/hal-01714054

### Conferences without Proceedings

[8] E. NTARYAMIRA, C. MAXIM, L. CUCU-GROSJEAN. *Ensuring data freshness for periodic real-time tasks*, in "the 12th Junior Researcher Workshop on Real-Time Computing", Poitiers, France, 2018, https://hal.inria.fr/hal-01900328

### Research Reports

[9] K. DIDIER, D. POTOP-BUTUCARU, G. IOOSS, A. COHEN, J. SOUYRIS, P. BAUFRETON, A. GRAILLAT. *Parallelisation efficace de larges applications temps-reel*, Inria Paris, June 2018, n$^{o}$ RR-9180, https://hal.inria.fr/hal-01810176

### Other Publications

[10] S. BEN-AMOR, L. CUCU-GROSJEAN. *Probabilistic parallel real-time tasks model on multiprocessor platform*, July 2018, pp. 1-2, RTSOPS 2018 - 9th International Real-Time Scheduling Open Problems Seminar, Poster, https://hal.inria.fr/hal-01956008

[11] S. BEN-AMOR, L. CUCU-GROSJEAN, D. MAXIM. *Response time analysis for precedence constrained and partitioned multiprocessor scheduled tasks*, October 2018, JWRTC 2018 - 12th Junior Researcher Workshop on Real-Time Computing, Poster, https://hal.inria.fr/hal-01957200

[12] L. FEJOZ, L. HAVET, A. DIDIER, B. VIAUD, A.-T. BUI LONG, T. D. NGUYEN, Y. OUHAMMOU, E. GROLLEAU, A. GOGONEL, C. MAXIM, L. CUCU-GROSJEAN, R. HENIA, L. RIOUX, N. SORDON, N. AYACHE, J. REHM. *Time4Sys – Integrating Timing Verification in your Engineering Practices*, December 2018, RTSS@Work 2018 - 39th IEEE Real-Time Systems Symposium Workshop, Poster, https://hal.inria.fr/hal-01957504

[13] E. NTARYAMIRA, C. MAXIM, C. FLORES, L. CUCU-GROSJEAN. *Towards temporal constraints in self driving cars*, July 2018, RTSOPS 2018 - 9th International Real-Time Scheduling Open Problems Seminar, Poster, https://hal.inria.fr/hal-01956016

## References in notes

[14] J. DENNIS. *First Version of a Dataflow Procedure Language*, in "Lecture Notes in Computer Sci.", Springer-Verlag, 1975, vol. 19, pp. 362-376

[15] J. FONSECA, G. NELISSEN, V. NELIS, L. PINHO. *Response time analysis of sporadic DAG tasks under partitioned scheduling*, in "11th IEEE Symposium on Industrial Embedded Systems (SIES)", 05 2016, pp. 1-10

[16] J. FORGET, E. GROLLEAU, C. PAGETTI, P. RICHARD. *Dynamic priority scheduling of periodic tasks with extended precedences*, in "IEEE 16th Conference on Emerging Technologies & Factory Automation, ETFA", 2011, pp. 1–8

[17] A. GOGONEL, L. CUCU-GROSJEAN. *Dispositif de caractérisation et/ou de modélisation de temps d'exécution pire-cas*, June 2017, n$^{o}$ 1000408053, https://hal.archives-ouvertes.fr/hal-01666535

[18] D. HARDY, B. ROUXEL, I. PUAUT. *The Heptane Static Worst-Case Execution Time Estimation Tool*, in "Proceedings WCET", Dubrovnik, Croatia, 2017

[19] H. HERBEGUE, H. CASSÉ, M. FILALI, C. ROCHANGE. *Hardware architecture specification and constraint-based WCET computation*, in "Proceedings SIES", Porto, Portugal, 2013

[20] H. HERBEGUE, M. FILALI, H. CASSÉ. *Formal Architecture Specification for Time Analysis*, in "Proceedings ARCS", Lubeck, Germany, 2014

[21] C. LIU, J. LAYLAND. *Scheduling Algorithms for Multiprogramming in a Hard Real-Time Environment*, in "Journal of the ACM", January 1973, vol. 20, n$^{o}$ 1, pp. 46-61

[22] D. MAXIM, L. CUCU-GROSJEAN, R. DAVIS. *Probabilistic schedulability analysis* , in "Handbook on Real-Time Computing", A. EASWARAN (editor), Handbook on Real-Time Computing, Springer, 2017, https://hal.archives-ouvertes.fr/hal-01666110

[23] D. I. MAXIM, R. DAVIS, L. CUCU-GROSJEAN, A. EASWARAN. *Probabilistic Analysis for Mixed Criticality Systems using Fixed Priority Preemptive Scheduling*, in "RTNS 2017 - International Conference on Real-Time Networks and Systems", Grenoble, France, October 2017, 10 p. [*DOI :* 10.1145/3139258.3139276], https://hal.inria.fr/hal-01614684

[24] C. MAXIM, A. GOGONEL, I. ASAVOAE, M. ASAVOAE, L. CUCU-GROSJEAN. *Reproducibility and representativity: mandatory properties for the compositionality of measurement-based WCET estimation approaches*, in "ACM SIGBED Review", November 2017, vol. 14, n$^{o}$ 3, pp. 24-31 [*DOI :* 10.1145/3166227.3166230], https://hal.archives-ouvertes.fr/hal-01666084

[25] D. POTOP-BUTUCARU, B. CAILLAUD, A. BENVENISTE. *Concurrency in Synchronous Systems*, in "Formal Methods in System Design", Mar 2006, vol. 28, n$^{o}$ 2, pp. 111–130