Cross-layer Design of Reconfigurable Cyber-Physical Systems

M. Masin*, F. Palumbo†, H. Myrhaug‡, J. A. de Oliveira Filho§, M. Pastena¶, M. Pelcat‖,
L. Raffo**, F. Regazzoni††, A. A. Sanchez‡‡, A. Toffetti[x], E. de la Torre[xi], K. Zedda[xii]
* IBM Research, michaelm@il.ibm.com, † U. d. Studi di Sassari, ‡ Ambiesense, § TNO, ¶ Science & Technology, ‖ INSA,
** U. d. Studi di Cagliari, †† U. d. Svizzera Italiana, ‡‡ Thales, [x] CRF, [xi] U. Politecnica de Madrid, [xii] Abinsula

***Abstract*—In the last few years, besides the concepts of embedded and interconnected systems, also the notion of Cyber-Physical Systems (CPS) has emerged: embedded computational collaborating devices, capable of sensing and controlling physical elements and, often, responding to humans. The continuous interaction between physical and computing layers makes their design and maintenance extremely complex. Uncertainty management and runtime reconfigurability, to mention the most relevant ones, are rarely tackled by available toolchains.**

In this context, the Cross-layer modEl-based fRamework for multi-oBjective dEsign of Reconfigurable systems in unceRtain hybRid envirOnments (CERBERO) EU project aims at developing a design environment for CPS based of two pillars: 1) a cross-layer model-based approach to describe, optimize, and analyze the system and all its different views concurrently and 2) an advanced adaptivity support based on a multi-layer autonomous engine. In this work, we describe the components and the required developments for seamless design of reusable and reconfigurable CPS and System of Systems in uncertain hybrid environments.

I. Introduction

Cyber-Physical Systems (CPS) and System of Systems (CPSoS) are complex systems composed of different interacting computing and physical entities that contribute concurrently to determine the behavior of the system as a whole. Computing layer and physical environment are tightly bound; therefore, such systems need to adapt, prospectively and autonomously, to rapid changes in the environment and in the system itself.

By their nature, CPS and CPSoS usually operate in uncertain environments and should satisfy multiple concurring and, usually, competing requirements regarding affordability, performance, safety, security, sustainability, etc. As a result, the design of CPS and CPSoS becomes inherently difficult, challenging, and time consuming. Promising ways of addressing this can be to follow new development methodologies, such as [1], and to use model-based design platforms, such as commercially available [2], [3] or academic [4], [5].

In the European Project CERBERO (Cross-layer modEl-based fRamework for multi-oBjective dEsign of Reconfigurable systems in unceRtain hybRid envirOnments) we are motivated by both approaches, and will build on previous results, coming from the DEMANES [6] and DANSE [7] projects, to tackle two important design and productivity issues of automatic design methodologies for CPS and CPSoS.

First, the intrinsic dynamic nature of CPS requires flexibility. Adaptation and reconfiguration have been deeply studied in the last decade, but standard solutions are not available yet. In particular, *self-reconfiguration and adaptation* have been acknowledged as key features for CPS operators dealing with faults management, but existing design frameworks rarely address them. CERBERO Project is meant to address this gap.

Second, despite their big promise (considering the claimed enhancement of and the declared speed-up), the existing model-based frameworks are not as popular as it could be expected. Modeling, maintenance, and interoperability overhead, especially with heterogeneous models, are, in fact, challenges not addressed in a satisfactory way [5]. Correctness-by-construction of reusable components implies efficient management of multi-physics, multi-abstraction and multi-fidelity heterogeneous models, capturing and optimizing cross-domain interactions. CERBERO intends to provide such support, designing all the needed components to enable seamless design and operation cycles.

The paper is organized as follows. Sect. II identifies the components of the design framework. In Sect. III we describe the envisioned validation plan. Finally, in Sect. IV we compare CERBERO approach with existing solutions and outline our plan for the incoming years.

II. Components of CERBERO Design Framework

Most properties and behavior of the whole CPS are emergent, i.e. they cannot be simplistically inferred from those of the individual components. The usage of *separation of concerns* to deal with heterogeneity, on the one hand, decreases design complexity but, on the other, it could miss important cross-domain interactions. We argue that the system could be sufficiently characterized by the following dominant aspects (or layers) when they are considered *together*:

- *computational* (or functional) *layer*: describing the functional behavior of the system, i.e., what it does,
- *physical layer*: describing the physical elements (devices) that compose the system and the uncertain hybrid environment where the system is, i.e., its surroundings,
- *communication* (or network) *layer*: describing which entities (*functional*, *physical* or both) exchange information and how they do it.

The layers could be in different (and mixed) levels of abstractions and hierarchy. Design of any CPS, no matter of its size, requires combining different modeling and analysis paradigms to effectively express the differences of the aforementioned

978-3-9815370-8-6/17/$31.00

layers. Moreover, because of inherent interdependence, the layers must communicate a lot. To handle these challenges CERBERO approach leverages on two main pillars:

- a *model-centric approach* to the scenario and system definition, where all the functional, physical, and network components as well as the environment are kept into consideration along with the properties that have to be addressed.
- a *multi-layer runtime adaptation strategy* for CPS, embodied within a high-level *self-adaptation engine* capable of reacting to external stimulus and autonomously adapting to the evolution of the scenario or constraints.

These two pillars, presented in Sect. II-A and Sect. II-B, respectively, contribute to determine the CERBERO *continuous design and operational framework* for highly interconnected systems presented in Sect. II-C.

A. Model-Centric Approach

Computational, physical and communication layers within CERBERO are going to be cross-optimized, taking into consideration that they are intrinsically concurrent, among each other and internally. To answer the current lack of a comprehensive modeling strategy for heterogeneous CPS, CERBERO intends to implement a component-oriented approach, with dedicated model-to-model mapping and synchronization interfaces with feedback loops. This strategy is meant to concurrently handle time-continuum and event-driven models that need to coexist, since they are representative of both the physical and the computing aspects of the system, while asynchronous, partially-ordered discrete actions or clock-driven time slots describe the communication layer. In order to reconcile these divergent models, and ensure interoperability and communication between components, we plan to incorporate the following elements in the CERBERO framework:

1) *Functional and non-functional requirements management:* Generic requirements (e.g., security, energy, dependability) and highly application-specific ones (e.g., the availability of charging points on EV network) are handled. By studying and defining adequate description languages and by providing a generic library of reusable *Key Performance Indicators* (KPIs) models, we intend to tackle adaptivity in different scenarios and models re-use in consecutive design and operational cycles. Such KPIs also offer an objective and quantitative analysis of the system.
2) *Entity/components cross-optimization and validation:* Dataflow Models of Computation (MoCs) provide precise semantics to express and, consequently, exploit the intrinsic computing problem parallelism. As demonstrated in literature [8], they are extremely suitable to model both hardware (HW) and software (SW) components at a high level of abstraction, and to perform trade-off analysis and components cross-optimization. On the other hand, state based stochastic algebraic and differential inequalities are suited to represent a large class of dynamic systems. The combination of these models is meant to input effective design space exploration (DSE) and multi-objective Pareto co-optimization of distributed systems (e.g. energy, memory, Quality of Service -QoS-, etc.). As an outcome of the project, we expect to provide libraries of such models, exploring also the network-related CPSoS parameters by mixing formal methods with event-driven and dataflow simulations.
3) *Flexibility and Adaptivity modeling and verification:* CPS are dynamically evolving systems required to adapt to variable external conditions. Runtime models, integrated in CERBERO-compliant platforms, intend to manage autonomous behavior, i.e., triggering reconfiguration in response to real-time sensed data, human inputs or varied system constraints. KPIs, evaluated by simulations and by executing component models in reconfigurable devices, are meant to estimate the costs and the effects of functional, technological and architectural parameters. They are integrated within the self-adaptation engine (see II-B) to feed a continuous evaluation of the system properties at runtime. Runtime performance/fault/energy monitors, in fact, are capable of providing self/environment-awareness to drive the embedded models. The models implement local smart decision-making mechanisms that propose (where and when needed) the most appropriate system (re-)configuration.

B. Runtime Autonomous Engine

CERBERO intends to implement cross-layer strategies for adaptation, based on runtime models, locally driven by sensed and processed data. The proposed strategy, to be implemented within a *self-adaptation engine*, is meant to provide *autonomous* runtime reconfiguration support, guided by functional scenario needs, changed QoS requirements, energy consumption criteria, sensor-processing analysis and reaction to unexpected and emergent behaviors.

Adaptation could be done in many levels: robust task scheduling policies reduce the need for reconfiguration on the first place; dynamic scheduling provides effective load balancing; functional tasks and requirements could adapt to changing conditions; communication and HW devices could dynamically reconfigure (i.e. components switched on/off or substituted) based on internal and/or external conditions. We intend to provide support and implementation of all these strategies, having identified two primary features for runtime adaptivity:

1) *Self-Adaptiveness:* Building on self-awareness, i.e., self-monitoring, and context-awareness, self-management is implemented to achieve predictable, autonomous response in uncertain hybrid environments. Dynamic resource management strategies are meant to trade off in real time between functional and non-functional requirements; while embedded estimation models are meant to support the decision-making process to dynamically select the optimal system configuration. Such dynamic models are meant to master the overall system adaptation (including HW, SW and network components) according to runtime sensed

(continuously retrieved by HW monitors or SW agents) or processed data.

2) *System Reconfiguration:* High-level models and intermediate representations are going to be exploited to enable the design of heterogeneous and coarse grain reconfigurable systems, regardless core types (e.g. ARM, FPGA, etc.) and based on industrial standards (e.g. LLVM, OpenCL), adding just-in-time compilation from some of their intermediate representations. Heterogeneous coarse grained computing devices provide massively parallel resources that can be used to enhance performance metrics. In this sense it may be extremely beneficial to jointly exploit coarse grained reconfigurable accelerators (limiting their inactivity with switch off policies) and dynamic partial reconfiguration to foster components reuse, replacing inactive slots by other elements that need to be executed. The Artico3 architecture [9] is to be used for HW acceleration, provides a method for runtime selecting a variable number of HW accelerators with adaptable module redundancy, performance and energy consumption. Artico3 will be adapted to other models of computation such as dataflow, combined with just-in-time compilation to provide functionality, performance and energy adaptation. Moreover, flexibility is achieved also by means of SW reconfiguration, which is guaranteed in a distributed way by the close cooperation of intelligent SW agents (that ensure optimal tasks execution according to the given constraints) and SW supervisors (that master the agents to decide upon tasks delegation and eventual migration). Work on the Parameterized and Interfaced Synchronous (PiSDF) DataFlow MoC, is envisioned for adaptive scheduling of parameterized processing tasks over heterogeneous multi-core systems.

C. Continuous Design Environment

To overcome the limits of current tools, CERBERO provides a continuous design environment enabling early-stage analysis, optimization and verification of functional and non-functional requirements, modifying CPS system design approach from a V to a ladder paradigm, as depicted in Fig. 1 and relieving designers from manual and complex HW/SW tuning phases.

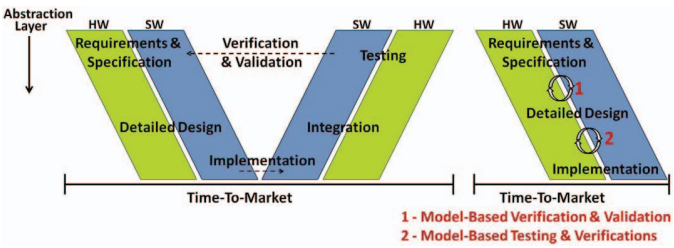

Fig. 1. CERBERO design approach.

The model-based approach (Sect. II-A) and adaptivity (Sect. II-B) pillars are supported by this design framework where all the properties and system characteristics are tackled concurrently right at the model-level to cross-optimize network and physical components, as well as, cross-configure the HW and SW layers. During the project lifetime, novel simplex-type algorithms are meant to be developed for the efficient optimization of dynamic systems described by state-based algebraic and differential inequalities with uncertain parameters, as well as, efficient representation of black box models of complex properties or verification tools. Here follow the primary framework features.

1) *Cross-Layer Optimization:* Ideal system partitioning and components cross-optimization, with emphasis on the reconfigurable HW and SW entities and their adaptivity agents/supervisors. The idea is adopting Mathematical Programming for simulation-based and stochastic control approaches to build cognitive CPS. Robust or stochastic multi-mode optimal control may recognize optimal conditions for mode change. High fidelity simulations and formal verification techniques providing feedback for optimization are meant to be incorporated in a holistic DSE cycle.
2) *Requirements analysis for maintenance and reconfiguration:* Verification tools and algorithms able to deal with requirements expressed with formal languages to support designers in rapidly identifying bugs, i.e. where the (sub-) system does not meet specifications, and to correct errors by means of (semi-) automated repair techniques, coping with the sources of complexity coming from CPS scenarios.
3) *Rapid Prototyping and Continuous Deployment:* Optimal-by-design system deployment. The goal is extending automated strategies and support tools to remove the need for double hard/soft competences, to bridge the SW productivity gap and relieve designers from the burden of characterization of the optimal node. In addition, system-in-the-loop simulation capabilities are featured to provide co-simulation of components models in virtual environments together with physical devices. This is a key-selling feature of our approach, fostering a gradual implementation of models into physical devices without losing their interaction with the whole system and environment. New physical components can then be verified and validated with the same set of stimuli and evaluation environment used during the modeling phase, ensuring effective system maintenance and incremental evolution.

CERBERO *continuous design and operational framework* is meant to be composed of a set of interoperable components, among which state-of-the-art tools are flanked by newly designed ones. Tab. I lists the set of tools currently representing the framework basis, emphasizing the planned advances/extensions necessary to tackle dynamic and heterogeneous CPS and CPSoS scenarios. One of the CERBERO outcomes is the definition of an integrated design environment, where all these components are interconnected and coexist.

III. Validation Plan

The effectiveness of the proposed approach will be assessed in challenging and diverse scenarios. We do not provide a one-fit-to-all unfeasible solution, rather we intend to exploit composability and incremental refinement cycles, leveraging on common modeling components and property libraries to be

TABLE I
COMPONENTS OF THE CERBERO DESIGN FRAMEWORK

Tool	Current Feature(s)	Foreseen Extension(s)
DynAA [10]	Agent-based analysis tool, combines features from system and network simulators. It provides: 1) Semi-integrated cross-layer modeling of computing and network components for self-adaptive networked systems. 2) Code generation for simulation purposes. 3) Special modeling constructors for large distributed system and for reconfigurable aspects.	1) More formal models of computation. 2) New system properties. 3) System-in-the-loop simulation.
PREESM [11] SPIDER RTOS [12]	They provide: 1) Parameterized dataflow specification of an application. 2) DSE and adaptive parallel code execution.	1) Support external events in simulations. 2) New Model of Architecture (MoA) to achieve simplicity, simulation speed and accuracy. 3) System- and SoS-level DSE.
AOW [13]	Cross-optimization component core. It provides: 1) Multi-objective and multi-layer architecture optimization. 2) Management of multiple metrics (i.e. HW/SW mapping, energy and data distribution, cost, failure rate, etc). 3) Support for Mixed Integer Linear Programming optimization.	1) Support for Robust Mixed Integer Continuous Linear Programming. 2) New system properties (i.e. reconfiguration and security).
Orcc [14]	Dataflow MoC Compiler. It provides several back-ends for HW and SW code generation.	1) Backend for runtime energy estimation on multi-core based on performance monitoring counters (PMC), for energy-aware SW reconfiguration purposes.
MarsPower[15] PAPI [16]	MARSPower, based on PMCs, provides energy consumption estimation in multi-core systems for reconfiguration purposes. PAPI manages PMCs.	1) Novel selection procedures of the training sequences to avoid over-fitting estimations. 2) Novel PMC-events filtering techniques.
ARTICo3 framework [9]	It provides: 1) Support for Dynamic Partial Reconfiguration (DPR) on FPGA. 2) Runtime post-implementation DSE. 3) Support for memory-mapped accelerators.	1) Support for dataflow MoC. 2) On-the-fly HW module composition. 3) Architecture-level dependability (e.g. hash-based temporal module redundancy to enhance fault tolerance).
MDC [17]	It provides: 1) Automatic deployment of coarse-grained multi-functional co-processors. 2) Dynamic power consumption management.	1) High-level profiling capabilities. 2) Support for combined Coarse-Grained Reconfiguration and DPR on FPGA.

specialized according to the use-case peculiarities. Common models include analysis *metrics* and *KPIs*, multi-mode *tasks*, *SW agents*, distributed and hierarchical *SW supervisors*, and reconfigurable HW and network *devices*. Data driven model discovery would support model customization to specific use cases. The evaluation procedure is meant to start from *use-case skeletons* (prepared assembling the above-mentioned common components and models) that will undergo a first compatibility test (to guarantee proper interconnection entities and databases) prior to the customization step. This allows achieving the following feedback on proposed libraries:

- substrate and network characterization, along with application/platform-specific code generation;
- compile-time mapping and partitioning to guarantee multiple critical constraints including (where and when necessary) dependability, security, power efficiency, high performance, size and cost;
- sensor and self-monitoring scenario-aware runtime support;
- continuous requirements analysis capable, by means of ad-hoc counter examples, to guarantee robust system maintenance and potentially trigger reconfiguration.

To demonstrate the portability of the proposed approach we have chosen three highly diverse test-cases, which span from a single embedded high reliability system to a highly networked one: a self-healing controlling device for the Mars Exploration Rover, a sub-sea and ocean monitoring SoS, and a highly networked Smart Travelling use case for Electric Vehicles.

A. Self-Healing System for Planetary Exploration

The objective of this use case is twofold. On one side it focuses on a single unique embedded CPS; while, on the other, it focuses on its integration with other systems of a planetary exploration mission. The use case focuses on the ExoMars mission, which will be the first mission to combine the capability to move across the surface and to study Mars at depth. The primary goal is to provide *self-monitoring* and *self-healing capabilities* by means of high performance sensor processing techniques, *triggering dynamic reconfiguration of the embedded computing system to overcome the failures* caused by the radiation or the harsh environmental conditions. The main controller device of the robotic units (e.g., arm and rover locomotion system) will be designed adopting commercial-off-the-shelf (COTS) solution and the capability to survive the Mars and space environment applying flexible, heterogeneous multi-core architecture, self-reconfigurable and with self-healing capabilities. Such support will ensure *long-term maintenance* of the system and will guarantee the possibility of *dealing with the stringent survival conditions and reliability constraints* of a robotic exploration mission. The CERBERO design framework is meant to master the following trade-offs.

1) Failure detection level and failure detection time vs. Monitoring logic overhead.
2) On-board Autonomy vs. Downtime/Reconfiguration Time.
3) HW ruggedization vs. Increased area/power.
4) Degree of autonomy and Type/number of correctable errors vs. System predictability.

The second objective focuses on a wider scenario taking into account the SoS typical of a planetary exploration scenario, which has different actors and system contributing to the final objectives, i.e. astronauts, supporting satellites, earth control station, exploration rovers and other support facilities among others. These systems are interconnected within a SoS; they usually have a static allocation of function and performance with poor or no self-reconfigurability or self healing capabilities. The main goal is to provide *self-monitoring and self-healing capabilities* by means of *sub-systems integration*. In particular, in case of anomaly, through the abstraction *hierarchy-based modeling*, we intend to enable a *cognitive*

work-based decision making process that investigates global solution involving all different systems, reaching then the overall mission target. The basic idea is to exploit the functionality of each of the systems involved in the mission for re-plan and re-configure the procedures to get the final mission objective even in case of a failure in a single system. The CERBERO approach guarantees efficient support for hierarchy-based SoS modeling and high-level analysis, to provide a decision making support both for astronauts and control engineers on earth for mission reconfiguration and re-planning.

B. Ocean Monitoring

A type of smart video-sensing unmanned vehicles with immersive environmental monitoring capabilities (i.e., marine eyeballs capturing videos/images of on-sea and subsea surroundings) composes the SoS representative of this scenario. Such marine robots can be *remote controlled* within wireless reach and visible sight, and capable of *self-operation and navigation*. Robots will be equipped with *new sensing and processing capabilities*, not only to navigate and operate the robot, but also for data analysis and information fusion. *Real-time processing and adaptation* is required to address the rapidly *changing environment conditions* in order to obtain or maintain positions on sea. From the computational point of view, a heterogeneous multi-core processor infrastructure is needed to integrate the sensors, wireless communication, and actuators, and further to manage vehicle navigation and information processing. The aim is for the electronic components of the robots to be *100% battery driven, solar and wind charged*, which would be particularly helpful in challenging recharging scenarios, i.e. the Arctic areas during winter time, or when communication with the vehicle has temporarily been lost. The CERBERO design framework aims to enable and support effective and cost efficient design of autonomous, adaptive and dependable marine robots by:

1) Cross-optimization of computation, storage, and network communication of the involved sub-systems.
2) Rapid prototyping/integration of autonomous vehicle with multiple sensors.
3) Usage of standardized communication within the interconnected SoS and the reuse of proven lower level SW drivers for sensors and actuators to keep development costs low.
4) Energy efficiency.

C. Smart Travelling for Electric Vehicle

The third scenario is the most networked one, being composed of different sub-systems:

- the *Electric Vehicle* with distributed communication capabilities, responsible also of monitoring and safeguarding the vehicle battery life;
- the *Person* who requires traveling and has normally multiple communication devices containing a Personal Agenda that determine his/her traveling requirements;
- the *Smart Home* to which the vehicle can connect to receive electricity and gather/provide data, managing localized home energy buffering;
- the *Smart Energy Grid* to which the vehicle can connect, from the home or other locations;
- the *Smart Mobility* provides mobility-aware functionality such as parking places, charge points, public transport, etc.;
- the *Smart Health* involves in-situ monitoring of health related parameters, suggesting/tuning related advice.

Due to the different involved heterogeneous concurrent subsystems, this scenario requires a *high degree of autonomy* and support for *adaptability* to cope with real life circumstances. Moreover, it requires to *integrate the distributed communication layers* of the different involved systems. The CERBERO model-based approach and its corresponding framework are meant to facilitate the design of such a complex CPSoS, where several dependencies and a plethora of highly different requirements have to be considered. For that, CERBERO will improve the state-of-the-art on (1) cross-layer and cross-domain optimization and validation of a multi-domain scenario; and (2) cross-verification and testing of physical and virtual (simulated) systems by means of the system-in-the-loop simulation.

IV. Final Remarks and Discussion

As already said, the design of CPS and CPSoS is entangled with complexity management at different levels. Dynamic behaviors and heterogeneity are top-priority ones, since their ineffective treatment can lead to performance failure and to difficulties in system development and maintenance. Adaptivity and heterogeneity management, by means of a complete design framework for reconfigurable CPS and CPSoS in uncertain hybrid environment, are the long-term mission of the CERBERO project. In Sect. IV-A we compare CERBERO with respect to the state-of-the-art approaches. Sect. IV-B defines our mid-term plan.

A. Beyond the State of the Art

Time-to-market and design productivity can be largely improved by the adoption of comprehensive and automatic design toolchains. Several methodologies and frameworks are available for CPS engineering (e.g., [2], [3], [4], [18], [19], [20], [21], [5]). All these tools offer support for different aspects of modeling and design, and, usually, simulation. Only few support optimization (e.g., [4], [5], [21]) and code generation (e.g., [2], [20]). None of them intrinsically supports CPS adaptivity and reconfiguration or in-the-loop simulation. CERBERO design environment, whose components cover all the above-mentioned features, exploits cross-optimization of components model-based design (as depicted in Fig. 1) to improve time-to-market and, in turn, productivity. Moreover, it is also capable of featuring support for heterogeneity that, despite a significant effort in this sense [22], is still an open issue. As an example, the envisioned seamless interoperability among AOW, DynAA and PREESM will allow the cross-optimization of heterogeneous multi-core nodes at the SoS level. PREESM will retrieve event-driven information from AOW or DynAA and combine it with dataflow information for cross-layer optimization (energy and latency aware) and for

ensuring computation reliability (deadlock-freeness, livelock-freeness, memory management). DynAA, Orcc, MARSPower, PAPI, ARTICo3 and MDC already partially address adaptivity and will be extended for in-loop simulation. Moreover, all the CERBERO compliant platforms are meant to be self-adaptive as explained in Sect. II-B.

With respect to other framework-oriented EU projects, CERBERO can be compared with INTO-CPS (http://into-cps.au.dk/) that aims at creating an integrated toolchain for comprehensive model-based design of CPS. It addresses modeling, design and verification, integrating existing industry-strength tools, based centrally around Functional Mockup Interface compatible co-simulation. CERBERO, as well as INTO-CPS, leverages on a model-based approach and on a comprehensive design environment, but it offers also runtime support for adaptivity and cross-layer optimization. With respect to this latter, AOW, the core component in charge of cross-layer optimization in the CERBERO design environment, intends to extend Measure-based Continuous Linear Programming [23] to support both linear and polynomial functions (more suitable for real-life problems), integrating objectives based on Robust Optimization with budgeted uncertainty and multi-objective Pareto-optimal hybrid solutions. The idea is exploiting Robust Mixed Integer Continuous Linear Programming to provide optimal system architecture and control policy, compliant with deterministic and uncertain environments as of CPS and CPSoS.

B. Mid-Term CERBERO Research Plan

The CERBERO consortiums vision is to extend and improve methods and tools to enable quick and cost effective design and deployment of interoperable, energy efficient, secure, and connected smart CPS and CPSoS devices. This is a long-term objective, which will continue long beyond the end of the project. To reach it, a set of mid-term goals have been defined and will be addressed as soon as the project begins:

- raising the level of abstraction in CPS design by defining:
  - highly reusable and customizable set of common analysis metrics and KPIs;
  - simulation and executable models of self-adaptive and reconfigurable components representing the different CPS and CPSoS layers;
- defining multi-layer (HW and SW) sensor-/data-driven reconfiguration strategies, implemented within a self-adaptation engine;
- defining continuous time cross-optimization strategies, executing runtime requirements trade-offs analysis and, when and where possible, system adjustments.

Along with these scientific goals, our research agenda includes also the definition of efficient model-to-model interfaces, framework integration strategies, and a preliminary definition of the skeletons of the use-case demonstrators (to enable quick prototype deployment and assessment). Finally, to guarantee a successful implementation of the CERBERO framework, a dedicated task provides the roadmap to a fully marketable version.

ACKNOWLEDGMENT

This project has received funding from the EU Commission's H2020 Programme under grant agreement No 732105.

REFERENCES

[1] NIST. (2016) Framework for cyber-physical systems, release 1.0. [Online]. Available: https://pages.nist.gov/cpspwg/
[2] Mathworks. Simulink/stateflow. [Online]. Available: www.mathworks.nl/products/simulink
[3] Modelica. Modelica/dymola. [Online]. Available: www.3ds.com
[4] L. Guo, Q. Zhu, P. Nuzzo, R. Passerone, A. L. Sangiovanni-Vincentelli, and E. A. Lee, "Metronomy: A function-architecture co-simulation framework for timing verification of cyber-physical systems," in *Conference on Hardware/Software Codesign and System Synthesis*, 2014.
[5] J. Sztipanovits, T. Bapty, S. Neema, X. Koutsoukos, and J. Scott, "The meta toolchain: Accomplishments and open challenges," no. ISIS-15-102, 2015.
[6] DEMANES. Design, monitoring and operation of adaptive networked embedded systems. [Online]. Available: http://www.demanes.eu/
[7] DANSE. Designing for adaptability and evolution in system of systems engineering. [Online]. Available: http://www.danse-ip.eu/home/
[8] D. Broman, E. A. Lee, S. Tripakis, and M. Törngren, "Viewpoints, formalisms, languages, and tools for cyber-physical systems," in *Workshop on Multi-Paradigm Modeling*, 2012, pp. 49–54.
[9] A. Rodriguez, J. Valverde, and E. de la Torre, "Design of opencl-compatible multithreaded hardware accelerators with dynamic support for embedded fpgas," in *Conference on ReConFigurable Computing and FPGAs, ReConFig*, 2015.
[10] C. J. van Leeuwen, J. M. de Gier, J. A. de Oliveira Filho, and Z. Papp, "Model-based architecture optimization for self-adaptive networked signal processing systems," in *Conference on Self-Adaptive and Self-Organizing Systems*, 2014, pp. 187–188.
[11] M. Pelcat, K. Desnos, J. Heulot, C. Guy, J.-F. Nezan, and S. Aridhi, "Preesm: A dataflow-based rapid prototyping framework for simplifying multicore dsp programming," in *European Embedded Design in Education and Research Conference*, Sept 2014, pp. 36–40.
[12] J. Heulot, M. Pelcat, K. Desnos, J. F. Nezan, and S. Aridhi, "Spider: A synchronous parameterized and interfaced dataflow-based rtos for multicore dsps," in *European Embedded Design in Education and Research Conference*, 2014, pp. 167–171.
[13] A. Zadorojniy, M. Masin, L. Greenberg, O. M. Shir, and L. Zeidner, "Algorithms for finding maximum diversity of design variables in multi-objective optimization," in *Conference on Systems Engineering Research*, 2012, pp. 171–176.
[14] Orcc. Orcc : Dataflow programming made easy. [Online]. Available: http://orcc.sourceforge.net/
[15] R. Ren, E. Juarez, C. Sanz, M. Raulet, and F. Pescador, "Energy estimation models for video decoders: reconfigurable video coding-CAL case-study," *IET Computers&Digital Techniques*, vol. 9, pp. 3–15, 2015.
[16] PAPI. PAPI: Performance application programming interface. [Online]. Available: http://icl.cs.utk.edu/projects/papi/wiki/PAPIC:PAPI.3
[17] C. Sau, P. Meloni, L. Raffo, F. Palumbo, E. Bezati, S. C. Brunet, and M. Mattavelli, "Automated design flow for multi-functional dataflow-based platforms," *Signal Processing Systems*, vol. 85, no. 1, pp. 143–165, 2016.
[18] SysML. Systems modeling language. [Online]. Available: www.sysml.org
[19] F. Mallet, "MARTE/CCSL for modeling cyber-physical systems," in *Formal Modeling and Verification of Cyber-Physical Systems, Summer School on Methods and Tools for the Design of Digital Systems*, 2015, pp. 26–49.
[20] Esterel_Technologies. Scade suite. [Online]. Available: www.esterel-technologies.com/products/scade-suite/
[21] Process_Modelling_Enterprise. Advanced process modelling platform. [Online]. Available: www.psenterprise.com/gproms.html
[22] J. Sztipanovits, T. Bapty, S. Neema, X. D. Koutsoukos, and E. K. Jackson, "Design tool chain for cyber-physical systems: lessons learned," in *Design Automation Conference*, 2015.
[23] G. Weiss, "A simplex based algorithm to solve separated continuous linear programs," *Mathematical Programming*, vol. 115, pp. 151–198, 2008.