Distributed Real-time Architecture for Mixed Criticality Systems

White Paper on Mixed-Criticality Research and Innovation

D9.3.2

Project Acronym	DREAMS	**Grant Agreement Number**		FP7-ICT-2013.3.4-610640	
Document Version	1.0	**Date**	September 30, 2017	**Deliverable No.**	D9.3.2
Contact Person	Gautam Gala	**Organisation**		TUKL	
Phone	+49 (0)631 205 2715	**E-mail**		gala@eit.uni-kl.de	

Version History

Version No.	Date	Change	Editor(s)
0.1	09.07.2017	initial draft	Gautam Gala
1.0	22.08.2017	First version	Gautam Gala
1.1	15.09.2017	Second version	Gautam Gala
2.0	18.09.2017	Ready for internal review	Gautam Gala
R1.0	29.09.2017	Submitted to EC	Gautam Gala

Contributors

Name	Partner
Gautam Gala	TUKL
Gerhard Fohler	TUKL
Simon Barner	FORTISS
Alexander Diewald	FORTISS
Roman Obermaisser	USiegen
Thomas Koller	USIEGEN
Daniel Gracia Pérez	TRT
Borislav Nikolic	TU Braunschweig
Marcello Coppola	ST
Miltos Grammatikakis	TEI
Alfons Crespo	UPV
Javier Coronel	Fentiss
K. Chappuis	VOSYS
Gebhard Bouwer	TUV
Gernot Klaes	TUV
Jon Perez	IK4-IKERLAN
Asier Larrucea	IK4-IKERLAN

Table of Contents

Summary

Safety-critical systems have become part of our daily life, for example, braking control in cars, controllers in aircrafts, wind turbines and trains, etc. A system failure or malfunction of these systems can cause death, serious injury to people, loss/severe damage to equipment/property, or environmental harm. An increasingly important trend is Mixed Criticality Systems (MCS), which combine safety critical components with components that need a lower level of assurance for failure. With increasing demands from the consumers, services like infotainment that need no certification are also required to be combined with safety critical components.

Design of MCS is becoming increasingly difficult due to adoption of COTS multicore and manycore devices. COTS multicore and manycore device designs are driven by large consumer segments like mobile phones, laptops, etc.; they do not take into account requirements posed by MCS in systems. With increasing demands for connected systems and the emergence of Internet based technologies like cloud computing, IoT, etc. and related technologies like Industry 4.0, autonomous cars, etc., the traditionally isolated MCS systems now have to be combine stringent real-time and reliability requirements with the need for an open-world assumption. As a result, they should not only consider safety but also security. The above trends, combined with closed hardware design, have made the process of certification for MCS difficult, expensive and time consuming.

This white paper outlines the research challenges and problems not solved in DREAMS wrt. MCS. It is a result of road-mapping process done during the DREAMS project with inputs from the project partners and the MCS community. It is meant to act as a guide for MCS research in the next 5–10 years.

1 Introduction

1.1 Motivation and Objectives

One of the key deliverables of the DREAMS Project "White Paper on Mixed Criticality Research and Innovation" is a roadmap for Mixed Criticality Systems (MCS). This white paper is the result of a roadmapping process that took place during the DREAMS project with involvement of project partners and the MCS community. It concisely describes the key research challenges and state-of-the-art in MCS and related application domains. Key challenges to be tackled in coming 5-10 years are described in this document. Moreover, it also gives an overview of problems not solved in the DREAMS project and innovations required to achieve the presented roadmap. The main idea of this white paper is to act like a guide to steer the research efforts by the MCS community.

1.2 Who Should Read this document?

This document is not only designed for specialists in the field of MCS but also those seeking to get a general overview of current research in MCS. We believe that anyone who is interested in key challenges in MCS and some of its application domains like Avionics, Wind-power, Healthcare, etc. in the next 5–10 years will find this document beneficial. We have also included some state-of-the-art for each key challenge to give an overview to the reader about the research current state. We also present some innovations in each key area and domain to give an idea to the reader about what is needed to achieve the presented roadmap.

1.3 Related Documents

A report [Inf] based on a workshop organized by the European Commission in Feb. 2012 identified the main research challenges in MCS design giving many recommendations defining European research priorities in the medium (3–5 years) and long term (5-10 years). It highlighted the developments in multicore technology and its adoption in MCS resulting in benefits for domains such as automotive and aerospace.

1.4 Essential Characteristics and Role in Future

Over the past few years, a trend of integrating components of different criticality levels onto a common hardware platform has evolved in real-time embedded systems. These criticality levels usually express required protection against failure for the corresponding system component to prevent catastrophic consequences; although, they can also include all forms of dependability (availability, integrity, . . .). These criticality levels are also know as Development Assurance Level (DAL) in the Avionics domain, or Automotive Safety and Integration Levels (ASIL) in case of Automotive. They need to be followed for certification.

Mixed Criticality Systems (MCS) are embedded systems which have applications of two or more distinct levels of criticality (for example, Low critical and High critical) sharing resources like CPU, memory, communication architecture, etc. MCS have received a lot of attention not only in research but also in industrial domains like Avionics, Automotive, Healthcare, Windpower, etc. For example, in case of Avionics or Automotive, there are some safety critical application functions like engine control, and many non-safety related functions like infotainment, air-conditioning, etc.

MCS are experiencing a massive paradigm shift to multicore systems as they need more performance and computing resources. Of recent, it has become quite evident that obtaining more performance from single core processors is not possible. Increasing clock frequency is becoming more difficult in embedded systems due to heating problems and energy requirements. Instead of using multiple single core systems, multicore systems have become popular as they can satisfy the required performance and provide reduced size, weight and power consumption (SWAP). But, the market of multicore systems is driven by the consumer market (for example, mobile phones, laptops, etc.), which defines the way these systems are designed. Commercial-of-the-shelf (COTS) multicores are not designed specially with MCS in mind. Hence, there is a need for MCS to adapt to these COTS multicore systems to harness their complete potential.

In addition, the trend of multicore is also evolving into manycore. Manycores are specialized multicores designed for a high degree of parallel processing, containing 10's or 100's or 1000's simpler, independent processor cores. In addition, some multicores have special purpose computing cores (for example for cryptographic algorithms, media processing, etc.) or a field-programmable gate array (FPGA) on the same system on chip (SoC), leading to heterogeneous multicore systems, adding on to the complexity of MCS design. In the past, MCS were mainly concerned by safety and reliability requirements as they were physically isolated and ran on dedicated hardware. But with the advent of Industry 4.0, Cloud Computing and Internet of Things (IoT), it has become important to combine stringent real-time constraints and reliability requirements with the need for an open-world assumption. Users increasing expectation for multimedia services is placing strong demands for the same to be available in MCS domains like air, rail and road transport. In the near future almost every consumer electronics device will be connected to an ecosystem for collecting and exchanging data, for example advance driving assistance, autonomous driving, etc.; and to provide services like payment systems, streamed content, etc. Experts estimate that the IoT market will consist of about 30 billion devices by 2020. Future MCS will be connected to cloud and other non-MCS systems, possibly from third-party vendors, where the trust is not very high. These systems are also called Open Distributed Real-time Embedded (ODRE) systems. Such MCS should not only consider safety but also security, esp. because there cannot be safety without security. For example, one of biggest threats faced by autonomous vehicles, an application domain of MCS, is vehicle cyber-security. One of the central challenges in vehicle security is that the various electrical components in a car (known as electronic control units, or ECUs) are connected via an internal network. Thus, if hackers manage to gain access to a vulnerable, peripheral ECU — for instance, a car's Bluetooth or infotainment system —, from there they may be able to take control of safety critical ECUs, like its brakes or engine, and cause catastrophic consequences.

Certification Authorities (CAs) demand the certification of MCS with strong confidence in the execution time bounds. As a consequence, CAs use conservative assumptions in the worst-case execution time (WCET) analysis which result in more pessimistic WCETs than the ones used by designers. Multicore and manycore systems, and open world assumptions add to the complexity of the MCS design and increase the certification cost, time and risks. Also, closed hardware design is making the certification of MCS more and more difficult. Furthermore, certification guidance from regulation authorities is still preliminary and not complete with respect to multicore systems.

Model-Driven Engineering (MDE) process enables to easily deploy applications onto a platform, and constitutes its interface to system designers and integrators. MDE can raise the level of abstraction in program specification and increase automation in program development in MCS.

These paradigm shift and upcoming trends have an impact on many aspects of MCS design. This paper details several key challenges that need to be tackled in next five to ten years, in order to improve mixed criticality systems. It is based on the road mapping process that took place during the DREAMS project with inputs from project partners and the MCS community. It is to act like a guide to steer the research efforts by the MCS community.

1.5 Related Projects

There are some preliminary set of projects to build upon in the area of safety-critical systems. The objective of **Distributed REal-time Architecture for Mixed criticality Systems (DREAMS)** project was to develop a cross-domain architecture and design tools for networked complex systems where application subsystems of different criticality, executing on networked multi-core chips, are supported. DREAMS delivers architectural concepts, meta-models, virtualization technologies, model-driven development methods, tools, adaptation/reconfiguration strategies and validation, verification and certification methods for the seamless integration of mixed-criticality to establish security, safety and real-time performance. Some other projects related to DREAMS and MCS are as follows:

- **Artemis CROSS Domain architecture (ACROSS)** project realizes a cross-domain multi-core chip with the service architecture defined in the GENESYS project, including a FPGA-based multi-processor system on a chip implementation of a time-triggered network-on-chip (TT-NoC), tailored middleware components and flexible, embedded tools.
- **Adaptivity and ConTrol Of Resources in embedded Systems (ACTORS)** project addresses the challenging problem of efficient design of embedded systems for complex and demanding high-performance applications. The project approach is to raise the abstraction level of the specifications and computing models, to include resource-constrained design exploration stages and real-time resource adaptation by developing the appropriate models and tools supporting the design from the specification down to the embedded implementation.
- **Automotive, Railway and Avionics Multicore Systems (ARAMIS)** project goal is the improvement of the operational safety of automobiles, trains and airplanes. Timing, determinism, influence on safety, real-time applications and certification are the major fields of research.
- **CErtification of Real-Time Applications desIgNed for mixed criticaliTY (CERTAINTY)** project aims to push forward the certification of real-time mixed critical embedded systems, a process currently challenged by the choices made at application design time about reliability and disturbances handling, which deals with the management of interferences between different functions of complex control software over the whole system.
- **Cost-Efficient Methods and Processes for SAfety-Relevant Embedded Systems (CESAR)** project was started to provide improved methods, tools and demands for embedded systems development in domains of Aerospace, Automotive, Automation and Railways. CESAR addresses the entire system engineering life cycle by improving its disciplines and implementing fundamentals for interoperability in a reference technology platform (RTP) as an integrated tool platform.
- **CRitical sYSTem engineering AcceLeration (CRYSTAL)** project project aims at improving the engineering process of embedded systems mainly by defining an Interoperability Specification (IOS) as a common backbone and means of integration among the various tools used in the design, engineering, implementation, testing and verification of embedded systems.

- **DynamIc Variability in complex Adaptive systems (DiVA)** project provides a new tool-supported methodology with an integrated framework for managing dynamic variability in adaptive systems. This is obtained by combining aspect-oriented and model-driven techniques in an innovative way.

- **Framework for Real-time Embedded Systems based on COntRacts (FRESCOR)** project is aimed at developing a framework that integrates advanced flexible scheduling techniques directly into the embedded systems design methodology, covering all the levels involved in the implementation, from the OS primitives, through the middleware, up to the application level.

- **Multi-cores Partitioning for Trusted Embedded Systems (MultiPARTES)** project is aimed at developing tools and solutions for building trusted embedded systems with mixed criticality components on multicore platforms.

- **Open VEhiculaR SEcurE platform (OVERSEE)** project aims to improve efficiency and safety of road transport by developing the OVERSEE platform, which aims to provide a secure, standardized and generic communication and application platform for vehicles.

- **REduced Certification COsts Using Trusted Multi-core Platforms (RECOMP)** project establishes methods, tools and platforms for enabling cost-efficient (re-)certification of safety critical and mixed criticality systems. Applications addressed are automotive, aerospace, industrial control systems, lifts and transportation systems.

- **Safety Certification of Software-Intensive Systems with Reusable Components (SAFECER)** project targets increased efficiency and reduced time-to-market by composable safety certification of safety-relevant embedded systems. The industrial domains targeted are within automotive, construction equipment, avionics, and rail.

- **Scalable and reconfigurable electronics platforms and tools (SCARLETT)** project implements the innovations in Distributed Modular Electronics (DME) concept to improve scalability, portability, adaptability, fault tolerance and reconfiguration capabilities in the current Integrated Modular Avionics (IMA1G).

- **Trusted computing Engineering for Resource constrained Embedded Systems Application (TERESA)** project defines, demonstrates and validates an engineering discipline for trust that is adapted to resource constrained embedded systems. Trust is defined as the degree with which security and dependability requirements are met.

- **Timing Model - TOols, algorithms, languages, methodology, USE cases (TIMMO-2-USE)** project addresses a significantly increased automation for more predictable development cycles in order to substantially reduce development risks and time-to-market, and to increase reliability, safety, robustness, and fault tolerance by a much higher degree.

- **TRustworthy Embedded systems for Secure Cloud Computing Applications (TRESCCA)** aims to lay the foundations of a secure and trustable cloud platform by ensuring strong logical and physical security on the edge devices, using both hardware security and virtualization techniques while considering the whole cloud architecture.

- **VARiability In safety-critical Embedded Systems (VARIES)** project aims embedded system developers to maximize the full potential of variability in safety critical embedded systems. The objectives of this project, with respect to safety critical embedded systems, are (1) to enable companies to make informed decisions on variability use in, (2) to provide effective variability architectures and approaches, and (3) to offer consistent, integrated and continuous variability management over the entire product life cycle.

- **VERification-oriented and component-based model Driven Engineering for real-time embedded systems (VERDE)** project aim is developing and industrialising a solution for iterative, incremental development and validation of real-time embedded systems in aerospace, software radio, railway and automotive domain.
- **SW/HW extensions for virtualized heterogeneous multicore platforms (vIrtical)** project aims the vertical and full development of the virtualization concept addressing the specific requirements for effective embedded virtualization. A virtualization-ready SoC platform and the associated programming models are developed, tackling all the system layers: applications, programming model, hypervisor and hardware.
- **Probabilistic real-time control of mixed-criticality multicore and manycore systems (PROXIMA)** project provides industry ready software timing analysis using probabilistic analysis for manycore and multicore critical real-time embedded systems, and will enable cost-effective verification of software timing analysis including worst case execution time.
- **SAFEPOWER** project goal is to enable the development of low power mixed criticality systems through the provision of a reference architecture, platforms and tools to facilitate the development, testing, and validation of these kinds of systems according to the market needs
- **Embedded Multi-Core systems for Mixed Criticality applications in dynamic and changeable real-time environments (EMC^2)** project finds solutions for dynamic adaptability in open systems, provides handling of mixed criticality applications under real-time conditions, scalability and utmost flexibility, full scale deployment and management of integrated tool chains, through the entire life cycle.
- **Design of embedded mixed-criticality control systems under consideration of Extra-functional properties (CONTREX)** project's main challenge is to guarantee timing, power, temperature, and reliability requirements by controlling (shared) resource usage and access on the execution platform. It has also considered extra-functional constraints right from the beginning, represented extra-functional properties in executable prototypes and included these properties into local and global scheduling and control decisions.
- **Open Platform for EvolutioNary Certification of Safety-critical Systems (OPENCOSS)** project provides the first European-wide open safety certification platform: an Open Platform for Evolutionary Certification Of Safety-critical Systems for the railway, avionics and automotive markets.
- **Safety And Security By Design For Interconnected Mixed-Critical Cyber-Physical Systems (SAFURE)** project targets the design of cyber-physical systems by implementing a methodology that ensures safety and security "by construction".

1.6 Document Overview

Chapter 2 to 5 represent some of the key domains of application for MCS:

Ch.2 Avionics,

Ch.3 Automotive In-Vehicle Networks,

Ch.4 Windpower, and

Ch.5 Healthcare.

Key research directions in MCS are represented by Chapter 6 to 12:

Ch.6 Real-time Systems and Scheduling,

Ch.7 Architecture,

Ch.8 Security,

Ch.9 Multi-core Chips,

Ch.10 Operating Systems and Hypervisors,

Ch.11 Certification, Safety and Dependability, and

Ch.12 Modeling and Tooling.

Each chapter is divided in three main blocks: Key Research Challenges, Problems not solved in DREAMS and innovations need for the Roadmap to be achieved. State-of-the-art for each of the corresponding challenge is also mentioned.

2 Avionics

2.1 Research Challenges

C.1 **Exploitation of multicores in safety critical systems**

The avionics domain has been replacing complex, fragile and heavy mechanical circuits by software (electronic) solutions to reduce the inconvenient of the mechanical circuits. An example of such, is the fly-by-wire solutions introduced in the civil market with Airbus A320, replacing the mechanical flight controls by computers based solutions (typically called LRUs). Traditionally each functionality required in an aircraft required one or more LRUs, but an LRU wasn't used for more than one functionality. This is so called Federated Avionics Architecture approach. However, as the number of required functionalities increases, that is no longer a solution. An aircraft like the Airbus A380 would require more than 100 LRUs (130 approximately) to implement the required functionalities [Iti07]. For that reason the Integrated Modular Avionics (IMA) architecture was introduced [EUR01]. IMA allows to integrate multiple functions in a single LRU with a single core processor, while ensuring the spatial and temporal isolation requirements. With the usage of the IMA for the design of the Airbus A380, the number of LRUs used was limited to approximately 100 [Iti07]. However, new and smaller aircraft will require more computing power to allow further integration or the deployment of more complex functionalities.

To enable these new aircraft designs without increasing the number of LRUs (or maybe even reducing them) multi-core processors are a promising solution. However, while they provide the required performance increase necessary, the interferences between cores [NPB+14] hinders their usage in safety critical products, as planes. In order to satisfy the temporal isolation requirements new *deterministic platform software solutions* [GJLR+15] need to be provided, while enabling an efficient exploitation of the multi-core processors. DREAMS addresses this issue by developing a *regulation software* solution, a distributed run-time WCET controller enhanced with quality of service approaches [KPR+14]. The controller enables the concurrent execution of critical and non-critical applications in a multi-core, ensuring critical applications deadlines while maximizing the non-critical applications utilisation of the processor. For that purpose the controller continuously monitor the critical application, and if a situation of danger is observed the non-critical applications are stopped until the finalization of the critical application time window.

State-of-the-art

S.1 The authors of [Fis14] propose a similar approach to the one proposed in DREAMS, however in [Fis14] during the execution of a critical application in one core all the other cores remain unused (or deactivated), so no concurrency is exploited when a critical application is executed. While that solution allows the deployment of critical and non-critical applications in a multi-core, the resources (the cores not used by the critical application) are not fully exploited.

S.2 In [YYP+13a] the authors propose a similar system to the DREAMS controller. It uses memory bandwidths controllers exploiting the processor performance counters to control the memory usage of the applications using the system.

C.2 **Fault-tolerance on distributed multi-core systems**

In order to ensure the reliability of the system stringent development rules as those defined in the avionics standards like the DO-178 [EUR92] for software development or the DO-254 [EUR00] for hardware development. However, a single system can always suffer a hardware or software failure which which can put the system (i.e. aircraft) in a critical situation. Among others redundancy solutions have been introduced [HM01] to ensure the safety of the system on systems failure occurrence, like: triple modular redundancy (TMR), triplex redundancy, hybrid TMR arrangement, or triplex voter-comparator. These solutions considered systems with single-core, with multi-core processors new solutions can be envisioned to reconfigure the system while ensuring the system safety. For example, failures in a core of a multi-core doesn't mean that the system using the faulty core needs to be considered as faulty, if the multi-core contains other cores that can be used to allocate the safety critical functions in the system.

DREAMS addresses fault-tolerance on distributed systems using multicores with a resource management solution capable of detecting core failures at runtime and reconfigure the system [DFG+16]. Core failure detection on a system (multi-core) is performed through a software monitoring service running on each core. On a core failure event, the system local resource manager is notified of the failure and a new schedule computed at design time is activated if possible (i.e. if the system has enough resources, cores, available to run the required functions). Moreover, in a distributed system, the system local resource manager reports its system status to a global resource manager, which can take global reconfiguration actions affecting multiple systems, to ensure that all the required functions (critical functions) are active.

To compute at design time the distributed system schedules in case of core-failure events a reconfiguration strategy and associated tool (GRec) is proposed. Reconfigurations graphs generated by GRec are used to configure the local resource managers and global resource manager to perform the reconfigurations at runtime.

State-of-the-art

S.1 In the SCARLETT project [PBB+12] an approach similar to the one introduced in DREAMS is proposed, the main difference being the resource management services architecture and that the authors in [PBB+12] only consider single-core processors.

S.2 The DIANA project [EJS+10] proposes a solution similar to those in DREAMS and SCARLETT [PBB+12]. While [PBB+12] proposes a centralized controller for the global reconfigurations, in [EJS+10] a decentralized controller is proposed. The approach proposed in DREAMS proposes a centralized controller, but at local level (a multi-core) reconfigurations can be done without being requested by the global resource manager.

2.1.1 Problems Not Solved in DREAMS

D.1 **Efficient integration of critical applications in multicores**

With the above mentioned deadline-overrun technique, see Section 2.1 C.1, a critical application can be combined with non-critical applications running in parallel in a multi-core. Multiple critical applications can be integrated in the system, the deadline-overrun technique doesn't address the issue of how critical applications can be executed concurrently.

State-of-the-art

This issue has been addressed in other works:

S.1 The authors in [DFG+14], [MNPGP16, MNN+17], and [TMW+17] propose to decompose critical applications tasks in three phases: the *acquisition* phase during which the task data is fetched from the main memory into a core local memory, the *execution* phase during which the task executes using only the local memory and the *restitution* phase during which task results. A simple, but stringent, constraint is put into schedules using this three phases task model: during the execution of a task *acquisition* or *restitution* phase no other cores can be executing other tasks *acquisition* or *restitution* phases. When following this approach, tasks (and thus applications) are ensured not to suffer interferences due to the usage of shared resources, as a single *acquisition* or *restitution* phase is ensured to use the shared resources at any moment.

S.2 The PROARTIS [CQnV+13] and the PROXIMA [CAA+16] propose an alternative development and design methodology based on probabilistic methods. Thanks to these approach, statistical methods ensuring the safe combination of critical applications in a multi-core could be used. However, to enable the usage of such methods processors developed presenting specific time behavior (random) must be used, making it difficult to apply to COTS[1] processors.

S.3 Agrawal *et al.* [AFF+17] propose a scheduling and runtime solution for systems similar to those studied in DREAMS (slot-based triggered-time systems) to allow the concurrent execution of critical tasks in a multi-core. Their solution is based on the definition of a scheduling taking into consideration the memory bandwidth requirements of the tasks executing in the processor. This even allows to assign the remaining bandwidth to non-critical tasks. A specialized runtime ensures that no task (specially non-critical tasks) uses more than the allocated bandwidth, stopping those tasks that have used the allocated bandwidth to avoid interferences not considered during the scheduling phase.

D.2 **Application state and fault-tolerance on distributed multi-core systems**

The DREAMS fault-tolerance mechanism implementation considers that the reconfigured applications are stateless or that can recover a state in the new configuration without compromising the system during the state recovering. However, state for non-stateless applications is completely lost on a reconfiguration between different nodes[2].

State-of-the-art

S.1 Redundancy (e.g., vote systems) solutions [HM01] provide this implicitly, as all the systems are always executing the application.

S.2 While projects as SCARLETT [PBB+12] and DIANA [EJS+10] have addressed the fault-tolerance on distributed systems issue, as in DREAMS they seem to not have addressed the application state.

[1]Component Off The Shelf

[2]If a reconfiguration moves an application from a core to another of a core the state is not lost, if both cores share the memory system.

2.2 Innovation for Roadmap

I.1 **Peripherals on mixed-critical systems**

While the subject of using multicores for safety-critical systems is an ongoing topic, few to none have addressed the usage of peripherals on such systems. For an example, in an aircraft a large number of functions behavior depend on an important number of sensors and external information. This information is typically acquired by the processor using different mechanisms (i.e. other than the memory subsystem) than those studied by the current literature, like specialized interfaces. The usage of these interfaces in multi-core systems introduce new sources of interferences that should be addressed.

I.2 **Combination of regulation and control mechanisms**

In the current literature a good number of regulation and control mechanisms [GJLR+15] have been introduced, as the deadline-overrun regulation mechanism introduced in DREAMS. Typically, regulation mechanisms address the combination critical applications with non-critical applications on multi-core processors, while control mechanisms address the combination of critical applications executed in parallel. The combination of both approaches could improve the utilization of the system and the performance of the applications running on it.

I.3 **Some other points regarding innovation for roadmap with respect to Avionics are:**

- Fault-tolerance solutions and redundancy
- Design of safety critical heterogeneous systems (FPGA and specialized circuits)
- HW/SW systems for mixed-critical systems
- Interference quantification for multicores

3 Automotive In-Vehicle Networks

3.1 Research Challenges

C.1 **An increasing amount of functionalities, and each of them with increasing processing and/or bandwidth requirements**

In the automotive domain, there are an ever-increasing demand for more powerful processing devices, and in-vehicle networks with higher bandwidths. In this chapter, the emphasis will be on the latter aspect.

The initial solutions for in-vehicle networks involved the bus-based communication. Such approaches have a straightforward support for the publisher-subscriber model, and several application specific standards emerged, such as CAN, LIN and FlexRay. The common benefit of all these approaches is the concept of predictable scheduling (e.g. via fixed priorities, time-division-multiple-access, slotted ring etc.). Routing the traffic across different networks can be done by intermediate gateways. However, these approaches have significant limitations, one of the most important ones being that they cannot cope with increasing bandwidth requirements of the automotive domain. Namely, all these approaches have an upper-bound on the transfer data rate (range between 100 kbit and 10 Mbit).

Future automotive systems are envisioned to have a vast number of very sophisticated functionalities, such as advanced driver-assistance systems, new generation of infotainment applications, and autonomous driving features. Implementing such functionalities within one system requires means to accommodate rapidly growing sensor traffic which can easily exceed the capacities of the aforementioned solutions (e.g. high resolution redundant image sensors). At the same time, such systems should also be able to accommodate complex low-latency traffic which is often of safety-critical nature.

State-of-the-art One viable solution to this problem is the switched Ethernet approach. The Ethernet networks are already well known in the domain of computer networks for their efficiency in terms of bandwidth. Their speed, at the present day, scales well with the growing technology demands (100Mbps $\rightarrow$ 1Gbps $\rightarrow$ 10Gbps $\rightarrow \ldots$). Another useful Ethernet characteristic is that it has open network capabilities, which is an important requirement for the open-world assumption. Finally, the Ethernet technology is widespread across other areas, such as avionics and industry, so there is already a substantial experience with using these networks, which can significantly accelerate the process of their integration into the automotive domain.

C.2 **Ethernet is envisioned as the backbone of the automotive network infrastructure. It is an imperative to employ data transfer strategies which are amenable to real-time analysis**

As already explained, CAN, LIN and FlexRay networks cannot be efficiently employed for the entire in-vehicle network of next-generation automotive systems. Nonetheless, they are still very efficient when applied to small sub-networks. Therefore, the most promising approach for the future in-vehicle networks is to have Ethernet as the network backbone, which connects sub-networks

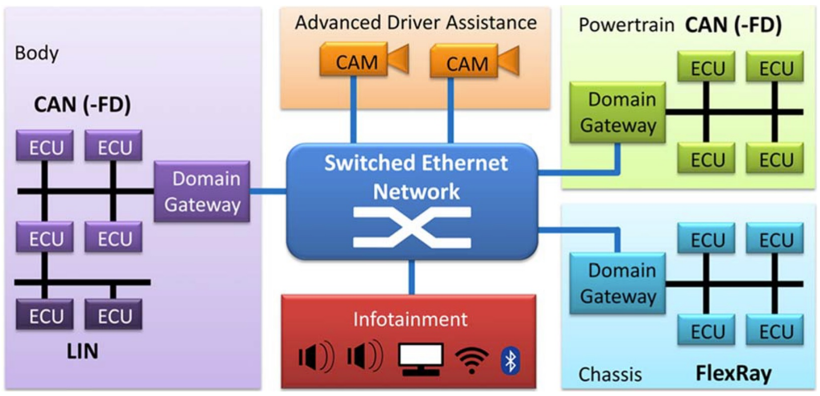

Figure 3.1: Example of in-vehicle network architecture

(CAN, LIN and FlexRay) into a unified system, where each of them presents the communication infrastructure dedicated to one functionality (e.g. powertrain, advanced driver assistance, infotainment). An example of this architecture is illustrated in Figure 3.1.

However, Ethernet was not designed with the real-time aspects in mind, and before it can be efficiently used in real-time systems, it is important to make it amenable to real-time analysis. The Time-Sensitive Networking (TSN) task group [tsn12] was founded with exactly that aim. This group works on a set of standards for time-sensitive traffic. One topic which attracts a lot of attention is the traffic shaping policy within network switches. The most well-known existing approaches are so called standard ethernet which includes static priority non-preemptive policy, and the AVB method, where traffic is, based on its criticality, classified into several categories which conform to different arbitration rules. In addition to that, the TSN group currently investigates several additional traffic shaping policies, such as Time-Aware Shaper, Burst-Limiting Shaper, Peristaltic Shaper, and Urgency-Based Scheduler. Moreover, the TSN group recently standardised the concept of frame preemptions [TSN16], which allows to achieve an ultra-low latency for high criticality frames.

As it is evident from the previous discussion, there are numerous ways and options to configure an Ethernet-based network. However, having more choices can is some cases be misleading, and usually presents an additional overhead when it comes to standards and inter-technology compatibility. Therefore, it is currently an imperative to perform a thorough and in-depth analysis and evaluation of existing techniques for real-time Ethernet, and proceed with the subset of the most promising ones. Additional aspects that need to be investigated are configurations of shapers (where applicable, e.g. AVB), topology selection, traffic classification and priority assignment (where applicable, e.g. standard Ethernet).

State-of-the-art The Ethernet technology has been extensively studied over the past several years. In this section, we will limit the discussion to the approaches related to real-time systems.

First, the formal analysis for the Ethernet standard (strict priority non-preemptive) and the AVB shaper was proposed [DTE12]. Then, the worst-case analysis method was proposed for the weighted round-robin arbitration mechanism [TDA+13]. The aforementioned worst-case analysis methods were gradually improved [ATED14, BAEP14, TAES14, TAE15, APB+17]. The worst-case analysis methods were also proposed for the later introduced Time-Aware Shaper and the

Peristaltic Shaper [TED15], as well as for the Burst-Limiting Shaper [TE16c] and the Urgency-Based Scheduler [SS16]. After that, the concept of frame preemptions was also analysed from the real-time perspective [TE16b]. Finally, the software-defined networking (SDN) concept for Ethernet was investigated [TE16a] as well as the gateway strategies for CAN-to-Ethernet networks [TSAE15].

C.3 **Automotive in-vehicle network as a collection of sub-networks connected with the Ethernet backbone**

With the envisioned automotive in-vehicle network architecture, where smaller sub-networks such as CAN, LIN and FlexRay are connected via an Ethernet backbone, one major challenge is to efficiently integrate all these technologies into a unique system. The most evident open question is how to organise the data transfer across different networks and how to configure the interfacing elements, often called gateways.

State-of-the-art As previously mentioned, the gateway strategies for CAN-to-Ethernet gateways have already been investigated [TSAE15], while gateways connecting other network types are likely to be studied in the near future.

C.4 **Safety aspects of in-vehicle networks**

Given that automotive domain falls into the category of real-time systems with numerous safety critical functionalities, it is an imperative to assure a high level of safety. The TSN group has proposed a proactive method towards transient transmission faults, which involves redundant paths. This approach has been standardised under the name 802.1CB - Frame Replication and Elimination for Reliability [TSN17a]. It allows to replicate the traffic within network elements and send it concurrently via redundant paths towards the same destination. It remains to be investigated how efficient this approach is, as well as are there many other techniques which can be used to mitigate the effects of transitional and/or permanent component/link failures. Finally, if it is possible to achieve a graceful degradation property and still meet some requirements when some of the system components experience a failure, remains an open question.

C.5 **Security aspects of in-vehicle networks**

Similar to the safety aspect, the automotive systems require a high degree of security. This aspect becomes more important and emphasised with the open-world assumption, where an extensive interaction of the system itself with the outside world is expected. At the same time, the increasing complexity of automotive systems implies that the same manufacturer cannot produce all functionalities, but instead some of them must be acquired from different suppliers. This raises a question whether all components of the system can be fully trusted, and also infers that the security aspects need to be addressed not only concerning the outside world, but also concerning the functionalities within the system itself. The TSN group proposed an approach based on stream filtering [TSN17b], which allows to prevent a network saturation. It remains to be investigated whether this approach is efficient, as well as are there some other security mechanisms that can also be applied. Finally, can Software-Defined-Networking (SDN) principles be used to enhance the system security is a valid research question.

3.2 Innovation for Road-map

I.1 The mix of different sub-networks of different type and purpose into a unique heterogeneous network system

The future automotive in-vehicle network is envisioned to be an heterogeneous system with small sub-networks based on CAN, LIN and FlexRay technology, connected via gateways to an Ethernet network backbone. The network topology and the configuration would be such that low latencies can be guaranteed for the high-critical traffic, and at the same time the best-effort service will be provided for functionalities that have significant bandwidth requirements. The configuration of network elements which represent the interface between different sub-networks (i.e. gateways) should be use-case dependent and heavily optimised for the given traffic, in order to provide the desired behaviour.

I.2 SDN is likely to be used in in-vehicle networks, and it will provide the means to implement safety and security features

The SDN functionality is likely to be used in the next generation of in-vehicle networks. Its concepts and application are well understood in different application domains, so the integration of SDN services into the automotive domain is expected to be relatively quick. Having a centralised monitoring entity offers lots of possibilities and efficient means to implement various safety- and security-related features. Given that enhancing safety and security are among the most important milestones in the development process of the future generation automotive vehicles, it is foreseen that advanced SDN systems are likely to be used to aid in that cause.

4 Wind Power

4.1 Research Challenges

C.1 **Increase system reliability and maintainability**

A modern off-shore wind turbine dependable control-system manages up to three thousand inputs and outputs, several hundreds of functions are distributed over several hundred nodes grouped into eight subsystems interconnected with a field-bus and the distributed software contains several hundred thousand lines of code [PGN+14a, PGN+14b].

The complexity of those systems challenges several dependability properties such as safety, availability, reliability and maintainability. The integration of additional functionalities also leads to an increase in the number of subsystems, connectors and wires increasing the overall cost-size-weight and reducing the overall reliability of the system. For example, based on domain specific field data from other domains it is considered that between 30-60% of electrical failures are attributed to connector problems [PGN+14a, SM99].

Therefore the achievement of the maximum probability that a system provides the specified service until a time, given that the system was operational at the beginning, is hampered by the high number of subsystems and networks. In the same vein, the complexity of such systems affect to the time interval required to repair them after the occurrence of a benign failure due to the high fault-potential amount of subsystems.

State-of-the-art

S.1 European research project MultiPARTES [MUL14] aimed to support mixed-criticality integrated architectures based on multicore and partitioning (e.g., hypervisor). This integration can increase system reliability by reducing the number of subsystems and associated wires and connectors [PGN+14a, PGN+14b].

S.2 As the usage of fans for device thermal dissipation is not considered due to poor reliability characteristics, thermal analysis and management becomes relevant for the development of next generation control platforms, in order to support the integration of multiple mixed-criticality functions in multicore devices [PGN+14a, PGN+14b]. CONTREX aims to enable energy efficient and cost aware design through analysis and optimization of real-time, power, temperature and reliability with regard to application demands at different criticality levels [CON14]. SAFEPOWER aims to enable the development of cross-domain mixed-criticality systems with low power, safety and security requirements [SAF17a].

S.3 EMC2 project's role is to explore the mechanisms for improving reliability on time transfer for industrial applications with a focus on the Smart-Grid market [EMC14].

C.2 **Integrated architectures**

The soaring demand for high performance and enhancing of the number of functionalities executed in today's embedded systems leads to the trend of moving towards integrated architectures

[Ham03]. System engineers aim at the integration of multiple functionalities with different criticality regarding safety, security and real-time on the same embedded computing platform. These systems are usually referred to as mixed-criticality systems [BBBT11, PGN$^+$14a, PGN$^+$14b].

C.3 **Single-core to multi-core transition**

Single-core processors were used in the last decades and are used nowadays to develop mixed-criticality systems. Partitioned single-core processors were implemented in different domains such as lift, automotive, wind power for developing safety-critical systems. Partitioning is achieved using virtualization solutions such as hypervisors, which allow dividing the processing core into smaller parts (partitions) with local and shared memory and exclusive access to peripherals. However, the needs of the industry advance fast, and the amount of the data required to be transferred gets bigger and bigger. Those requirements limit the applicability of single-core processors such as their available resources are limited and are not scalable. Therefore, faster and more capable processors are required to accomplish the market request.

Multi-core and many-core processors are the natural transitions of single-core processors such as they provide high-performance capabilities that increase due to microarchitecture advances. Those processors provide benefits in terms of lower power consumption, which according to Pollack's law, it is linearly proportional to the increase in complexity, and lower size-weight-cost. They also provide better scalability, availability, maintainability and reliability.

C.4 **Be able to meet safety, security and real-time**

Mixed-criticality systems allow implementing applications with different criticality levels such as real-time, safety and security into the same system. Therefore, those systems should be able to meet criticality-related constraints such as time independence, spatial independence, diagnosis. For instance, a non-safety partition could access to the memory region of a safety partition and modify the data, or it could affect the temporal behaviour of the safety partition, taking too high a share of the available processor execution time.

In addition to this we should consider that nowadays and in the future: 'there is no safety without security' [Per17].

C.5 **Attest multi-core computing systems**

Multi-core devices are prompt to interference due to resource sharing (memory, peripherals, processor time) and the communication between the elements necessary to achieve the overall goal design (partitions). Those issues challenge the attestation of systems based on a multi-core processor.

IEC 61508 Annex F recommends a set of "Techniques for achieving non-interference between software elements on a single-computer" [IEC10c]. However, those techniques are not at all applicable in multi-core devices, such as one resource can be shared among the elements of the system. Therefore, the need for new or extended measures and diagnosis techniques is identified in this project.

C.6 **Attest product families**

A product line is a range of similar products that are developed by the same company. Individual products in such a product family are similar, but still slightly different. The current state of the art is to attest individual systems, where if safety, security or real-time aspect changes, the entire system should be re-certified. There are, however, attempts to achieve incremental certification where only those parts that need recertification are reassessed when a product is enhanced or comes with a new version. This task goes one step beyond incremental certification into modular and product-line certification.

State-of-the-art

S.1 In OPENCOSS research project a model based assurance of safety-critical product lines is defined [KHM+16].

C.7 Integrate model driven engineering

Model-driven engineering (MDE) focuses on creating and exploiting domain models, representing the knowledge and activities of a particular application domain. MDE can be integrated with safety life cycles (e.g., IEC 61508) in order to increase the development efficiency, cost reduction and reduce the probability of human systematic errors.

State-of-the-art

S.1 OPENCOSS research project integrates the goal structuring notation (GSN) MDE to represent safety cases [OPE14].

C.8 State of the art and next generation SoC

The number of transistors in the integrated circuits doubles proximately every two years. This observation entails the increase of the capabilities and complexity, which impacts on the development and certification cost of next-generation system-on-chips (SoC). The performance of those systems increases due to microarchitecture advances. The performance is roughly proportional to the square of the increased complexity.

Moreover, reducing the on-chip power consumption has become a key challenge in SoC domain. The continuing trend for ever increasing functionality, performance, and integration within SoCs is leading to designs with power issues (dissipation). These systems require expensive packaging, heat sinks, and a cooling environment, which lead to additional matters related to the availability and maintainability of future developments.

C.9 Integration of programming paradigms

Modern off-shore wind turbine dependable control-system manages several hundreds of functions which are distributed over several hundred nodes grouped into different subsystems interconnected with a field-bus and the distributed software that contains several hundred thousand lines of code [PGN+14a, PGN+14b]. Those systems allow to integrate different programming paradigms (e.g., C, C++, HDL) as they enable to integrate processing systems and programmable logic into the same system architecture.

C.10 Digitalization of the wind turbine

Digitalization has revolutionized the proprietary infrastructure and the control and management of dependable embedded systems. Patent documents can now be obtained in a digital version. Physical records are being digitalized to ease the access. Furthermore, digitalization enables the remote control of systems such as wind-turbines, monitoring and controlling their behaviour and reacting to any discrepancies detected.

Nevertheless, the digitalization has associated risk factors, such as security. The security threat potential in the digital era needs to shift from securing network perimeters from safeguarding data propagated across systems, devices, and the cloud.

This challenge needs to consider the integration and evolution of IIoT, e.g., QoS communication with cloud infrastructure, cloud computing, delocalization of some real-time and safety partitions, cybersecurity.

4.1.1 Problems Not Solved in DREAMS

D.1 **Support real-time, safety and security**

The wind turbine system defined in European project DREAMS focuses on safety and real-time. Security was not considered in the case study definition.

D.2 **Attest multi-core mixed-criticality systems**

Although in DREAMS we develop a set of generic and reusable patterns applicable to hypervisors, COTS multi-core devices and mixed-criticality networks, further solutions could be generated to tackle the remaining issues detected in [LAO+16, LMB+17].

D.3 **Attest wind turbine product families**

In DREAMS we generate a safety argumentation scheme for IEC 61508 compliant system, which aims to decrease the cost and time required for developing and certifying a wind turbine. This scheme defines the safety-related arguments of a wind turbine and the evidences that support those arguments. However, the implementation of the argumentation system has not been validated in an industrial safety project or formal certification assessment.

4.2 Innovation for Road-map

I.1 **Product family modular safety argumentation scheme**

The attestation of a product family is an expensive and time-consuming process analysed in some research projects (e.g., OPENCOSS). DREAMS research project contributes a generic and reusable modular IEC 61508 compliant safety-argumentation scheme based on a wind turbine product family. Also, this argumentation scheme supports variation points from standards. It supports safety-domain-specific standards such as it is based on IEC 61508, which is the main standard for E/E/PE functional safety, and security and real-time standards such as it was generated to support further domain criteria (e.g., ISO 26262, ISO 13849). In addition, it supports variation points from requirements. The modification of one requirement of one of the components that compose the product line does not affect the overall design as it is defined considering modularity methodology. Modularity enables the subdivision of a system into smaller parts (modules), which can be independently generated and certified.

I.2 **Generic and reusable cross-domain mixed-criticality patterns**

In this project, we analyse the IEC 61508 safety standard, and we identify some improvements regarding the techniques applicable to partitioned and networked multi-core systems. As the result of that analysis we define, implement and generate a set of generic and reusable patterns to solve remarkable issues of those systems.

I.3 **Multicore / Manycore with virtualization for the development of next generation high-computing mixed-criticality solutions for for Wind Turbines**

Innovation roadmap already described in Sections 2, 3 and 9.

I.4 **Wind Turbine digitalization and Industry 4.0**

The digitalization of wind-turbines with the integration and evolution of Industrial Internet Of Things (IIoT), opens a new innovation path between the integrated architectures to be deployed in the wind turbine and the external cloud architecture to be deployed integrating technical challenges such as QoS communication with cloud infrastructure, cloud computing, delocalization of some real-time and safety partitions and cybersecurity.

5 Healthcare

5.1 Research Challenges

C.1 **Real-Time ECG Monitoring**

Prolonged real-time ECG monitoring is needed for improved detection of diseases. Most industrial products, e.g. AliveCor, BodyGuardian, LifeMonitot, NowCardio, and PhysioMem, capture, process and transmit ECG data to a server for offline analysis by specialists. In addition, research on wearable monitoring and arrhythmia diagnosis concentrates on detection capability and focuses on Android and IoS smart apps. Android smartphone is often used to capture, analyze and visualize ECG for alerting the patient wearing the device in real time (but not the doctor).

State-of-the-art Transmission of ECG signals to a remote monitoring center for offline processing by medical personnel has been previously considered, with very limited work on hard and soft real-time communication. In respect to soft real-time, there are limited results on a) ECG data packetization and selection of TCP parameters during network transmission and b) on-the-fly ECG signal analysis and visualization of annotated patient data for real-time monitoring by physicians or clinicians. These aspects, treated in DREAMS, make our research on soft real-time ECG processing innovative.

5.1.1 Problems Not Solved in DREAMS

In DREAMS, we have evaluated real-time performance of ECG analysis for detecting non-fatal arrhythmias and visualization versus non-critical video processing when memory and network bandwidth regulation techniques are in place using both in-hospital (Linux-based), and out-of hospital (smartphone Android-based) scenarios based on a mobile wearable cardiac pulse sensor (ST Bodygateway). Although results are encouraging, the parameter space has not been fully explored in respect to memory, network and CPU bandwidth requirements and limits. Other issues relate to the support of the Bluetooth protocol on open source RTOSes for embedded systems, e.g. FreeRTOS. These issues affect system predictability of mixed criticality real-time embedded systems (inside and outside the hospital) supporting on-the-fly ECG analysis and visualization together with other applications (e.g. on a hospital media server), such as video decoding or streaming.

5.2 Innovation for Road-map

Extensions to supporting full diagnosis of different types of arrhythmias, with online monitoring of other important physiological data (heart rate, respiration rate, physical activity level and body position) on-demand in real-time is interesting. This requires additional paths, either from Doctor App to Patient App to program the BGW to send other biometric data (e.g. respiration or accelerometer), or from Cloud server to Patient App to allow patients to monitor their own heart rate in real-time, informing them and physicians to adjust medication, or call emergency services in case of a significant cardiac event.

6 Real-time Systems and Scheduling

6.1 Research Challenges

C.1 **Quantifying timing interferences of shared resources in multicore systems**

The use of multicore systems is rapidly increasing as execution platforms for real-time embedded systems because of their better performance/energy ratio, reduce infrastructure costs, and ease of management. Also, The field of embedded systems in experiencing a trend towards integrating multiple applications on a single multicore platform. However, the interference in shared resources in multicore platforms like caches, shared buses, memories, etc., poses several problems, and it is yet to see widely adopted commercial solution. It may severely reduce the performance of tasks executed on the cores, and increases the complexity of timing analysis and/or decreases the precision of its results.

The issue of timing analysis on multicore platforms is because of interference delays as a result of conflicting simultaneous access to the shared resources. These issues were not present in single core systems, or not at the same scale. For example, the latency of a memory access in single core depended only on the type of memory being accessed and whether it was cached or not. But in case of multicore, the latency of a memory access also depends on whether the same memory region is being accessed by other cores or not, as multiple simultaneous access may lead to contention; only one request can be served at a time, while others have to wait. The authors in [NPB$^+$14], for example, demonstrated that the latency of a single memory store operation can increase by a factor of 25.82 in a Freescale P4080 multicore platform when the number of active cores increases for 1 to 8. Thus the determination of worst-case execution time is complex, and the estimation is much higher than the observed average/best-case. Methods for scheduling tasks and resource access that help in reducing pessimism of WCET analysis are required.

State-of-the-art Many techniques use Time-Division Multiple Access (TDMA) like arbitration mechanism where resource access is provided to tasks in statically assigned time slices. These techniques provide high predictability, but yield poor resource utilization since unused slices are wasted [KHMF13]. For example, Schranzhofer *et al.* [SPC$^+$11] divide the task execution in phases, and tasks are only allowed to access shared resources in certain phase while making sure no other task are simultaneously in resource access phase. Bonal *et al.* [BCNP12], divide the task execution in sub-tasks and sub-tasks further in execution and communication slices. Shared resources access is only allowed in a communication slice. Some solutions involve using custom hardware like the predictable DRAM controller [VY15]; but it is not commercially viable to implement in hardware for a small market segment corresponding to real-time systems. Thus, restricting use of Commercial Off The Shelf (COTS) multicore platforms. Joint analysis methods like those consider by Chattopadhyay *et al.* [CKR$^+$12] are also popular, but have large computational complexity as they analyze the program flow on all cores considering shared resources. Several approaches, for instance [NPB$^+$14], propose resources reallocation based on information derived from monitoring the utilization, e.g. the memory accesses. Agrawal *et al.* [AFF$^+$17] and similar works consider interference sensitive WCET (is-WCET), that computes offline the execution time of applications considering the memory access from applications that can execute in parallel.

C.2 Simultaneous execution of mixed-criticality applications on multicores of a single platform

The trend of multicore system also applies to safety critical domains, like Avionics and Automotive, where applications of several criticality levels execute on a single multicore platform. These criticality levels express the required protection against failure for the corresponding applications and system. For example, in case of Avionics, these criticality levels are also know as Development Assurance Level (DAL), and they are needed to be followed for certification. To execute such mixed criticality applications on a single multicore requires strict temporal and spatial isolation. Achieving this isolation is not that straight forward in multicore systems, esp. because many shared resources are required for performance and cost efficiency. Also, if applications of different criticality levels are allowed to access the shared memory in a random manner, then an application of lower criticality accessing a shared resource can block the access of any high criticality task that is executing in parallel. This can severely effect the response time of the high criticality applications. It is also difficult to quantify the impact on response time because of limited knowledge about the behavior of low criticality applications and some hardware resources like the shared-bus.

State-of-the-art As an initial step, safety-critical single-core avionics applications are ported to a multicore by preserving the original schedule as well as the source code while executing it on only one core. Historically, safety-critical real-time systems have been implemented using a cyclic executive (CE). This approach has been extended for multicore platforms by Burns *et al.* [BFB15]; the execution is coordinated on all cores so that the frames are released at the same time, and only applications of same criticality are allowed to execute concurrently. In DREAMS, we propose running a high-criticality task in parallel to low-criticality tasks on a multicore platform by using DREAMS resource management services (RMS); the execution of low critical task(s) is suspended on detection of a possible deadline overrun by the DREAMS RMS monitor. We further propose techniques to improve the system usage and performance of the low critical applications by using quality of service management schemes [DFG+16].

C.3 Incremental system extensions

Modern systems constantly need to update and upgrade in order to provide more features and to fix problems with the previous selves. For safety critical systems though, upon an update of a part of the system, the complete system needs to be verified & validated [ENNT15] (or certified) as per safety standards (IEC26262, for instance) in order to guarantee correct operation after deployment. If a system is designed without considering potential future extensions/modifications, the cost and efforts for certification of the system might prevent its deployment for the application. On the contrary, when the potential extensions are envisioned during the design phase of the system, only a number of modified sub-systems and the overall system functioning need to be certified [ENNT15].

To enable future extensions/modifications, the design and scheduling tools need to allow modification in the design without modifying the existing functionality. There exist a number of tool design approaches which intrinsically support such functionality by freezing the existing design and using the left-over resources. However, other approaches might not be as flexible for changes. In other words, the selection of the tool design approach affects the possibility for future extensions.

State-of-the-art The Xoncrete tool in the DREAMS tool-chain supports incremental scheduling by freezing a partition or a set of tasks and utilizing the left-over resources to schedule new tasks.

Similarly, a modification in a partition can be made without affecting the rest of the partitions (thanks to ARINC style partitions). Another approach was proposed by ¡GA¿ et al. [?] using compositional scheduling framework, however this approach might lead to potential bandwidth loss as mentioned by Lackorzynski et al. [LWVH12].

Note that, other than the inter- and intra-partition schedule, a re-configuration of other artifacts (e.g. on- and off-chip network schedule, virtual links, ports, modes of operation, etc.) might also be required upon an update. Although other tools in the DREAMS tool-chain do not support incremental development of a system [BDM+17b], but due to the flexible design of DREAMS meta-models and tool-chain such a feature can be added with minor efforts [BDM+17b].

C.4 Safety and Security-aware scheduling

In the past, these real-time systems were mainly concerned by safety and reliability requirements as they were physically isolated and ran on dedicated hardware. But many safety critical systems nowadays are networked or distributed and are moving towards a open-world assumption. For example, the trend of Internet of things (IoT) is coming to automotive systems; passenger comfort and infotainment features continue to progress through the advancement of in-vehicle networks and connectivity of the automotive system with its environment. These systems can no longer be bounded by static system structures but need to take into account components entering and leaving at runtime. The challenge here is not only to consider distributed and networked real-time systems, but also allow dynamic system structures and open world assumptions without compromising safety and reliability requirements of the systems.

In addition, these systems have also become an promising target for active and passive attackers. In the worst case, compromising a few components can bring down the entire system. An important challenge is ensuring the continuous unmaintained real-time operation of the system while guaranteeing safety and reliability requirements.

State-of-the-art In DREAMS project, we developed mechanism for detecting core failure, and provide fault-tolerance via reconfiguration in distributed system by using DREAMS RMS services[DFG+16]. In case of a core failure in multicore systems, critical applications are locally reconfigured in order of priority. Remaining applications that cannot be schedule, are reconfigured on another multicore platform of the distributed system if possible. During the DREAMS project, we developed security mechanisms to ensure the security of systems RMS communication as it handles critical system information and scheduling of tasks in the distributed DREAMS system [GKGP+16] [KGP+16]. Four different security levels are proposed which can be used as per the security requirement of the individual RMS communication channel. The same security mechanisms can also be used by applications that need secure communication.

Until recently security was an afterthought in the design of real-time systems. Most existing real-time systems don't consider security issues. Some works consider timing inference based security attacks as real-time systems, esp. Time-triggered systems that execute the same schedule, are vulnerable to them due to their inherent predictable nature. Krüger *et al.* [KVF17] propose an online mechanism to obfuscate the schedule through randomization while ensuring deadlines are met.

It is fairly well understood that the use of shared resources in a platform can lead to information leakage without the need for explicit communication between tasks. Mohan *et al.* [MYPB14] propose method to prevent information leakage through scheduling constraints and cleaning states of shared resource, e.g.: flushing caches between runs of applications with different security requirements. These methods consider simplistic threat model and impose large number of constraints. In addition, cleaning up states of shared resources like cache can severely impact performance of the system. Pellizzoni *et al.* [PPY+15] also present a model for preventing information leakage in hard real-time systems showing how to capture security constraints as relationships between tasks and a new vendor oriented security model for information leakage.

6.1.1 Problems Not Solved in DREAMS

D.1 **WCET and Scheduling Interdependence**

The cyclic dependency between WCET and scheduling makes it difficult to define a feasible schedule.

State-of-the-art Groesbrink *et al.* [GAdSP14] propose single-Core equivalent virtual machines to make WCET independent of scheduling. [RNH+16] proposes co-estimation of WCET and schedule by modeling cache behavior, and accounting it in the schedule.

D.2 **Running critical applications simultaneously in multicore systems**

In DREAMS it is possible to run low criticality applications simultaneously with a high criticality application on a multicore by using deadline-overrun methods. DREAMS RMS suspended the execution of low critical task(s) on detection of a possible deadline overrun by the monitors. But, it is not possible to run two high criticality applications simultaneously in DREAMS. Efficient integration of critical applications in multicores is detailed in 2.1.1.

D.3 **Fault-tolerance while maintaining application states**

In DREAMS applications are assumed to be state-less or can recover their execution state at time of reconfiguration on their own without compromising the system during the time of state recovery. This is explained in depth in is detailed in 2.1.1.

D.4 **Multi-mode Support for Networks**

DREAMS considers a single network mode which combines all possible network schedules.

State-of-the-art Different modes/configurations of network schedule can provide more flexibility upon resource failure. Multi-mode support is present in Time Triggered Protocol hardware by defining two mode types: Immediate and Deferred [KNH+97]. Heilmann *et al.* [HSF16] propose to integrate all messages from all modes in a single schedule is to overlap the scheduled transmission intervals of messages belonging to distinct modes. Since only one mode can be active at any point in time, the risk of collision between messages with overlapping transmission intervals is avoided.

6.2 Innovation for Road-map

I.1 **Dynamic system structure and open-world assumption**

Real-time systems should consider system components that can be added or removed at run-time. Up till now, real-time systems consider closed world assumption and only execute in their isolated island. But with emergence of IOT technologies and cloud computing, real-time systems need to have a open-world assumptions. Also, Many projects have consider dynamic system structure and open-world assumption, but none of them address stringent real-time requirements and reliability properties in such systems.

I.2 **Understanding constraints imposed by safety criticalities in multicores**

To execute safety critical applications on a multicore requires strict temporal and spatial isolation. Achieving this isolation is not that straight forward, esp. because many shared resources are required for performance and cost efficiency. A better understanding of safety criticalities in multicores is required.

I.3 **Identify/define security and safety constraints needed**

We need to identify what constitutes a potential threat to safety and security in real-time systems. An important questions that arise with respect to safety criticality systems is that if we can still considered criticality levels just for safety or whether we should also include security in criticality levels. If security criticalities are considered, then how should they be addressed, and what differentiates them from safety criticalities? For example, a high critical application may only need integrity, authenticity and access control for communication, but a low criticality application may need confidentiality in addition as it communicates sensitive information.

I.4 **Mixed criticality systems on current Hardware, especially in COTS platform**

Design special hardware for small market segment corresponding to real-time systems is not a commercially viable option. Hence, the focus should be on developing methods to use COTS platforms for mixed criticality systems. Hardware architectures like Xilinx Zynq that combine existing processor with FPGA on a single SoC may provide a cost effective solution for adding specialized hardware architecture to certain resource for give real-time guarantees.

7 Architectures

7.1 Research Challenges

C.1 **Open world assumption in combination with large scale, real-time and dependability requirements**

In recent years the field of embedded systems has evolved towards novel application areas that combine stringent real-time constraints, reliability requirements and the need for an open-world assumption. These systems are also called Open Distributed Real-time Embedded (ODRE) systems [Yu09]. System architectures need to cope with dynamic system structures, where components enter and leave at run-time, interact among each other in variable setups and realize different global application services. Examples are Ambient Assisted Living (AAL) systems for elderly care [AGRS06], networked medical devices and health management systems [AGRS06], applications for electrical power distribution [SWG+08] and command/control systems [Gri06, PM99]. These systems are based on an open-world assumption where new components are integrated at run-time in order to dynamically realize emerging global services. At the same time, reliable operation and support for stringent real-time requirements are essential to support closed-loop control and guaranteed response times. For example, physicians need to dynamically integrate medical devices into an in-home AAL system for emergency treatment, while ensuring predictable response times and reliable interaction with in-home devices (e.g., medical sensors) and remote sites (e.g., hospital).

State-of-the-art. Architectures for large-scale open systems were addressed in the scope of System-of-System (SoS) and Internet-of-Thing (IoT) initiatives (e.g., CASAGRAS, ebbits, IoT@Work, IoT-I, IoT-A). These initiatives resulted in models, protocols and processes for semantically integrating the IoT into mainstream enterprise systems and support interoperable real-world, on-line end-to-end business applications.

However, reliability and real-time capability are today not addressed in these existing architecture. An open research challenge is the development of a distributed embedded system architecture for constantly evolving and dynamic SoS with support for verifiable real-time and reliability properties. The system architecture needs to support reliable closed loop control with stringent real-time requirements for applications.

C.2 **Modeling and management of intended and undesirable emerging services**

Components interact in embedded systems in order to realize emergent services (e.g., optimal and dependable energy distribution in smartgrids) that cannot be provided by any of the components in isolation [KHF+15]. However, systems can also exhibit unintended emergence that results from the interactions of the components. An example is an energy-grid blackout which due to feedback-loops between producers and consumers.

Two types of emergence can be distinguished, namely weak emergence and strong emergence [Cha06]. A system-level phenomenon is strongly emergent with respect to its components, when the system-level phenomenon is not deducible from the low-level component behaviors. In case of weak emergence, the system-level phenomenon arises from the component behaviors although the system-level behavior is unexpected given the principles at the component level.

Understanding, predicting and managing desirable and undesirable behaviors is an important research challenge. Research objectives include analytical methods for predicting weak emergence, monitoring services for recognizing weak/strong emergence and simulation models for predicting emergent behaviors in real-world systems. In addition, mechanisms for managing emergent behaviors are required in order to facilitate desired emergence and avoid detrimental ones.

State-of-the-art. Emergence has been extensively studied in different research disciplines [Cha06, BH08]. The study of emergence in the context of cyberphysical systems has also become a major focus in the area of cyberphysical systems-of-systems [Bon16]. However, the state-of-the-art does not provide methods for predicting and managing intended and undesired emergence in cyberphysical systems.

C.3 Diagnostic services for decreasing required redundancy and associated cost in safety-critical fail-operational systems

In safety-critical applications, an embedded computer system has to provide its services with a dependability that is better than the dependability of any of its constituent components. Considering the failure-rate data of available electronic components, the required level of dependability can only be achieved if the system supports fault tolerance. An important step in fault treatment is diagnosis, which determines the causes of failures in terms of localisation and nature. Using a root-cause analysis, anomalous behaviors and states are traced back to the originating fault. A component can fail because of a design fault in software or hardware, a transient or permanent physical hardware fault or an operator mistake. From a phenomenological point of view, the distinction between these faults is not simple. For example, transient hardware faults may have the same manifestation as the sporadic activation of a design fault.

Active diagnosis exploits this diagnostic information at run-time for fault isolation and online error recovery. For example, possible reactions to permanent faults include the entering of degraded service modes and the migration to another suitable resource (e.g., another processing element for computations, another physical channel for a message). A research challenge is the development of safety-critical systems with convincing safety arguments based on active diagnosis, while minimizing the addition of redundancy to the system for fault tolerance. For example, reconfiguration triggered by active diagnosis promises to reduced cost compared to Triple Modular Redundancy (TMR).

State-of-the-art. The state-of-the-art offers a wide variety of methods for diagnosis including model-based diagnosis (e.g., [Con11, PLt00, Ise06]), rule-based inference methods [Rei87] and probabilistic diagnosis methods (e.g., [Mil87]). To deal with incomplete knowledge of systems, machine learning approaches for diagnosis were developed such as neural networks [Elh11], RBF networks [Fri94], genetic algorithms [PTH$^+$90], neuro-fuzzy methods [DHR93] and pattern matching and clustering [KSM08]. Another important class of diagnosis methods is based on fault trees [KM03], including parameterized fault trees [BFGP03], fuzzy set partitioning with fault trees and decision tree classifiers [Cer10]. An example of active diagnosis is the classical PMC model [PMC67], which was extended to deal with transient faults and distributed analysis [BDM93]. Online diagnosis algorithms for runtime isolation and recovery were introduced by [WLS97]. A sequential approach dealing with multiple faults in dynamically changing systems is described in [ORM], where the D-matrix [SB07] addresses incomplete or imprecise information. Fault-Detection, Fault-Isolation and Recovery (FDI) using Bayesian inference and dynamic decision networks with parameterized fault trees, was used for an on-board architecture for active diagnosis [PCR11]. Another approach for active online diagnosis [JKT08] uses residuals which are generated by parameter estimation, taking into account the parametric perturbations in a non-linear model. Another online fault diagnosis approach for non-linear systems uses Kalman filters with a set of samples to approximate the current belief state and keeps the distribution up-to-date with new observations [HD03].

7.1.1 Problems Not Solved in DREAMS

D.1 **DREAMS addressed hierarchical multi-cluster systems, but not systems-of-systems with an open-world assumption.** DREAMS supports end-to-end channels over hierarchical, heterogeneous and mixed-criticality networks. DREAMS introduces gateways for a system perspective of mixed-criticality applications combining the chip-level and cluster-level. On the one hand side, DREAMS supports uniform communication between heterogeneous and mixed-criticality networks. Gateway services enable this horizontal integration at the cluster-level across different off-chip communication networks with different protocols (e.g., TTEthernet, EtherCAT, etc.), different reliabilities (e.g., fault-tolerant networks with media redundancy and active star couplers, low-cost fieldbus networks). In addition, gateway services between NoCs and off-chip networks enable vertical integration through the seamless communication in hierarchical networks respecting mixed-criticality safety requirements.

However, DREAMS has not addressed systems with an open-world assumption, where components enter and leave at run-time, interact among each other in variable setups and realize different global application services.

D.2 **Anticipated intended and undesirable emerging services are considered, but not unanticipated ones.**

DREAMS considers anticipated intended and undesirable emerging services as part of the development methods and the design space exploration. The development tools of DREAMS deal with emerging timing and safety properties in networked multi-core chips as well as mixed-criticality product lines. The DREAMS simulation environment allows fault injection and timing evaluations of networked multi-core chips by co-simulating the software execution environment, the NoCs and the off-chip networks. However, the prediction and management of unanticipated emerging services was out of the scope of the DREAMS project.

D.3 **DREAMS provides specific fault recovery techniques, but no systematic and complete solution for diagnostic services based on different failure assumptions in order to decrease cost in fail-operational systems.** DREAMS has introduced hierarchical resource management services, which can be used for fault recovery and increased reliability by means of adaptation. DREAMS provides an effective baseline for active diagnosis in fail-operational mixed-criticality systems, but further research on a systematic and complete solution for diagnostic services based on different failure assumptions is required.

Research questions are predictable and assured methods for identifying faults based on observed failures and anomalous behaviors are required. Different types of faults including software faults, transient and permanent hardware faults, faulty inputs and imprecise specifications must be covered, while achieving a guaranteed diagnostic coverage.

7.2 Innovation for Road-map

I.1 **Coordination services for concurrent, distributed and incremental scheduling and resource allocation.** One expected innovation are incremental, distributed and concurrent resource-allocation algorithms for cyberphysical systems. The search for a feasible schedule must be computed incrementally upon the introduction of new applications. The distributed computation of the schedule using the different constituent systems considers the lack of global knowledge and control, while also reducing the overall scheduling time. Concurrent scheduling activities are supported to deal with the uncoordinated and possibly simultaneous introduction of multiple applications.

I.2 **Integration of semantic techniques and machine learning into embedded system architectures.** The dynamic composition of components in new setups, which were not analyzed and planned at development time requires methods for knowledge management in order to ensure the semantic compatibility of the integrated components. At the same time, systems should minimize the manual development and configuration efforts by employing machine learning as a basis for self-organization, self-configuration, self-optimization, self-healing, self-protection, self-explaining, and context awareness.

I.3 **Diagnostic services combining real-time guarantees, assured detection coverage and expressive power.** Active diagnosis can only avoid the replication of components in fail-operational safety-critical systems, if guarantees with respect to detection coverage and diagnostic latencies are available. At the same time, the expressive power of the diagnostic models must be sufficient to support a wide range of faults including software faults, hardware faults and interaction faults.

8 Security

8.1 Research Challenges

C.1 **Linkage of safety and security**

Security is becoming a major subject in MCSs, as there is no safety without security. Security vulnerabilities directly affect the safety and dependability of the system. The relationship between security, safety and dependability is analyzed in [ALRL04]. There are several examples showing the importance of security in safety critical systems. In an aircraft, the access to the flight control system may be possible through its entertainment system [Eva15]. [KCR+10] showed the possibility to manipulate components of the car, e.g., the radio, the instrument panel cluster (speedometer, etc.), the engine and the brakes. In [CMK+11], the software of a car was manipulated and then they accessed the car remotely. [MV15] showed a remote attack against a car without modifying its software. Only with the remote access, they were able to send messages on the CAN and affect physical systems. Similar to safety requirements, security requirements have to be considered already during the design phase of a system. Trying to integrate security services after the design of a system will not lead to a secure system.

State-of-the-art There are guidelines and best practices for different industrial branches, such as automotive or power supply, e.g., for smart grids. [Nat16] describes cybersecurity best practices for modern vehicles. Guidelines for smart grid cybersecurity are described in [PB14].

C.2 **Integrating MCS in an open-world scenario**

The integration of MCSs in an open-world scenario (Section 7) raises new security challenges for MCSs. In this scenario, a MCS will be conected to other (non-MCS) systems. This includes the connection to the cloud, or application fields such as Industry 4.0 / Intelligent Manufacturing / Smart Factory. These systems may influence the MCS somehow. Therefore, the MCS has to be protected from malfunctions of the open world system, or from attacks form the open world against the MCS. The continuous working of the MCS has to be guaranteed.

State-of-the-art To separate a MCS from the open world and to control the traffic from, and to, the MCS, mechanisms such as firewalls can be used. In addition to firewalls, [BGP+15] describes the approach of a secure and trusted cloud.

C.3 **Connection sensors and actors**

MCSs relies on a large number of sensors and actors. The integrity of the functionality of these components is important for the operativeness of the system. Not only the integrity of the components themselves, but also the components of these components is important. An attacker might be able to replace some of the sensors and actors. Sending false data to the MCS (sensors), or acting in a wrong behavior (actors), compromises the MCS. If sensors or actors are connected through the Internet, e.g., IoT devices, such attacks are even easier.

State-of-the-art Not only the integrity of the sensors and actors and their provided data or performed actions has to be ensured, but also the authentication of these components has to be performed. The approach of [KGP+16] and [GKGP+16] can be extended and then applied to sensors and actors. Privacy and security in Internet of Things and wearable devices analyzed in [AWHJ15].

C.4 **Security for Real-time Scheduling**

Security for real-time scheduling is described in Section 6, C.4.

8.1.1 Problems Not Solved in DREAMS

D.1 **Trusted Hardware and Software**

To ensure that only trustworthy software is executed, a root of trust has to be established. Both the hardware and the software has to be trustworthy. Starting with a trusted hardware, a secure boot only executes verified software, e.g., if the hardware directly starts the hypervisor, the hardware has to check the integrity and the authentication of the hypervisor. The hypervisor in turn has to check every executed software component. [AFS97] describes a secure and reliable bootstrap architecture. [SAB15] analyzes the ability of a trusted execution environment. A approach for platform integrity verification for a mobile trusted module is presented in [KJK15]. Secure and trusted clouds are presented in [BGP+15].

D.2 **Hardware security issues from inside**

Often, the full functionality of a 3rd party intellectual property is not known by the purchasing manufacturer. A malicious 3rd party intellectual property can attack the system from inside. [ACR14] shows a approach to mitigate the treat of components being connected to the NoC.

8.2 Innovation for Road-map

I.1 **Further understanding of security and safety coherences in MCS**

There are several examples that show the impact of vulnerability on an MCS [Eva15, KCR+10, CMK+11, MV15]. The safety related impacts of attacks have to be analyzed further to allow a better understanding of the coherences.

I.2 **Identifying/defining the security requirements of a MCS when it is connected to the open world**

The integration of MCSs in an open-world scenario raises new security challenges for MCSs. This includes the connection to the cloud, IoT devices, or Industry 4.0 / Intelligent Manufacturing / Smart Factory. The impact of malfunctions in the open world on the MCS has to be identified and appropriate security requirements have to be defined.

9 Multicore Chips

9.1 Research Challenges

C.1 **Multicore Resource Management (Memory, Network, CPU)**

There is growing interest in the real-time operating systems community on extending and modifying the Linux scheduler to support different system-level bandwidth management policies that can help differentiate among rate-constrained and best effort traffic sources on multicore systems-on-chip. The extensive Linux scheduling infrastructure is not so easy to grasp, just by reading the code, not only because of the multiple extensions for flexible scheduler configuration, but, also, because of the complexities and the amount of code for load-balancing of multicore (SMP) and multi-chip (NUMA) machine configurations. Coordinated efforts in this domain are needed to provide state-of-the art solutions that target significant domains, such as embedded systems and server domain.

State-of-the-art Our work has extended existing memory bandwidth regulation policies, i.e. genuine MemGuard Linux kernel module (LKM) by providing improved adaptivity through EWMA prediction and a violation free operating mode for rate-constrained flows. The resulting extensions (MemGuardXt) follow a highly modular approach that allows using MemGuardXt in multiple instances. By applying the same algorithmic approach as MemGuardXt, we have also developed a network bandwidth regulation LKM (called NetGuardXt) running over netfilter to regulate incoming or outgoing traffic per IP at packet or throughput-level. Finally, notice that per-process (or more accurately per-group) CPU bandwidth regulation can be examined on top of CFS Linux scheduling policy by configuring cgroups.

9.1.1 Problems Not Solved in DREAMS

Within DREAMS, we have studied MemGuardXt/NetGuardXt effects in correlation to real-time by examining a mixed-criticality use case on a hospital media gateway prototype (Zedboard with two ARM Cortex-A9 cores). The gateway runs soft real-time ECG analysis on one core (relying on our extensions of open source PhysioNet packages), and stores video-on-demand traffic for subsequent video streaming on the second core. By examining different NetGuardXt/MemGuardXt configurations, we have shown that fine-grain control of network/memory bandwidth can improve ECG processing, especially when violation free mode is used. In this use-case, the so-called DREAMS healthcare demonstrator, several open questions relate to optimal control of memory, network and CPU resources at process-level.

9.2 Innovation for Road-map

Holistic resource scheduling strategies that globally manage memory, network and CPU bandwidth in Linux-based distributed embedded systems either in coarse grain (per-core), and especially fine-grain (per-process) are currently in a primitive stage. In this effort, existing work on kernel-level memory bandwidth regulation (MemGuard, MemGuardXt), NetGuardXt, as well as CPU bandwidth are only independent steps. A future aim towards a refined system framework, supported by directives and tools that can optimally approach global resource management at-process level is sought. Towards this goal it is necessary to develop alternative Linux schedulers (or policy options in existing schedulers) that allow regulation of rate-constrained and best-effort memory traffic at process- instead of core-level. New policies combining memory, network and CPU bandwidth regulation are also interesting.

In addition, high-level system-level design models and exploration tools that examine interactions between system-level memory bandwidth regulation policies that operate at core- or process-level and low-level (hardware-supported), highly accurate system-on-chip and network-on-chip QoS protocols can help classify and evaluate interactions between memory, network and CPU scheduling policy options for time critical and rate-constrained traffic flows at different abstraction levels. Notice that the level of accuracy of these global solutions in system monitoring, as well as timing requirements for actions must be examined.

10 Operating Systems and Hypervisors

10.1 Research Challenges

C.1 **Virtualization techniques to deal with full virtualization (HW support)**

Virtualization refers to the creation of a virtual machine or partition that acts like a real computer with OS, but executing the software applications separately from the underlying hardware resources. Among other objectives, virtualization is intended to support system partitioning and to protect the execution time and memory space of each application.

The processing capability that multi-core embedded systems can reach permits to run multiple applications on a single shared hardware platform even if some of them are time-critical applications. Under these circumstances the need for integrating applications into the same hardware combining time-critical and non-critical applications is strongly recommended taking into account the increment of computation power of current processors.

Hypervisor based systems have demonstrated their ability to provide basic properties and mechanisms to execute several applications, including its operating system, in a common hardware. These properties deal with the spatial and temporal isolation of the software partitions (application and operating system) running on top of the hypervisor [SCO08].

Virtualization techniques can based on partial or full virtualization. While partial or paravirtualization [DFH+03] requires to adapt the operating systems on top of the hypervisor, full virtualization can execute guest operating systems without modification in a transparent way. Most of the current processor provide hardware support for virtualization that can be used by the hypervisor to ease the virtualization to the guest applications including the operating systems.

On the other hand, the reactivity and the performance of the virtualization layer are crucial for Mixed Criticality Systems. Using paravitualization, these issues can offer reasonable values. However, full virtualization techniques must be improved to reach the desirable parameters.

State-of-the-art

S.1 Poper and Goldberg [PG74] introduced a set of requirements to support efficient virtualization. The most important requirement established that critical instructions that could affect the correctness of system functioning (e.g. change of sensitive registers and memory locations, such as a clock register or interrupt registers) shall trap and pass control to the partitions, enabling them to emulate the desired effect in the guest OSs. Today several processors provide an Instruction Set Architecture (ISA) which meets Poper and Goldbergs requirement, including Intel VT-X, AMD-V, ARM and PPC architectures). This hardware support for virtualization leads to reduced overhead and footprint of virtualization software and improved performance. Most of current processors distinguish between user and privileged modes when executing instructions, and include a Memory Management Unit (MMU) with the dual objective of translating virtual addresses to physical addresses, and preventing unwanted memory accesses. Some processor architectures, such as Intel VT-d, AMD-Vi and SPARC V8, also support device and Input/Output (I/O) virtualization. An I/O Memory

Management Unit (IOMMU) enables partitions to directly use peripheral devices through DMA I/O bus and interrupt remapping. Besides, several hardware extensions for device virtualization improve networking and I/O throughput for partitions (e.g., PCI-SIG I/O, network virtualization of Intel VT-c, single-root I/O virtualization SR-IOV). However, the integration of I/O into an architecture with time and space partitioning remains a research challenge. E

S.2 Hybrid solutions for MCS virtualization should be more appropriated. Low level services such as partition context switches, interrupt management, etc., can be supported by the full virtualization based on hardware provided by the processor. Additionally, some partition can require to know that the systems is composed by several applications (partitioned) and avoid the transparent virtualization. The definition and efficient implementation of the two components at hypervisor level will permit to reach the timing and performance requirements of MCS. In [NLY+11], it is analysed the advantages and drawbacks of para-virtualization versus full virtualization.

S.3 In [HE16], a review of the evolution of the L4 microkernel is analysed with special emphasis in the new principles for virtualization such as minimality, generality and efficiency.

C.2 **Communication services virtualization**

As a result, handling distinct levels of criticality in software systems from different industrial sectors can be a typical scenario in the future of embedded systems engineering. However, even when the adoption of multi-core and distribution provides significant benefits to mixed-criticality systems, some challenges remains open from the communication perspective. MCS can involve several hardware platforms with different communication devices. In DREAMS, STNoC and TTEthernet communications have been integrated. Some of the solutions for communications in partitioned systems are based on IO Servers where a partition is in charge of the communication drivers and provides services to other partitions. In that case, the communications are not virtualized and a partition handles the IO devices.

State-of-the-art

S.1 A number of virtualization techniques for on-chip and off-chip communication networks are available. Time-triggered networks use Time-Division Multiplexing (TDM) to establish virtual communication links where components cannot interfere with each other in the value and time domain. TTNoC [(Ed11] and AEthereal [GH10] are examples of on-chip networks for time and space partitioning. A major challenge here is the seamless virtualization of resources at chip and cluster level. For example, the access to a remote I/O resource located on another chip should be carried out via gateways involving different on-chip and off-chip networks (i.e., vertical integration), and possibly gateways between different types of off-chip networks (i.e., horizontal integration).

S.2 Increasing the communication responsiveness of partitioned systems. Traditional approaches to build ARINC-like partitioned systems often suffer from a significant loss of performance in communications as a result of time partitioning [PG16]. This loss of responsiveness in communications may become an impediment in taking advantage of the benefits of partitioning in modern complex embedded systems from different industries, as they are increasingly networked and might even require global connectivity. In this context, enhanced communication mechanisms are needed to develop distributed partitioned systems suitable to satisfy different safety, real-time and communication requirements. The use of multi-core platforms opens up the possibility of executing several partitions in parallel, which may serve as a basis for the development of these new communication mechanisms.

S.3 Integration of the communication services at hypervisor level will provide fast access to the devices and reduce the latency of the communications. The main drawback is the certification of the hypervisor for a wide spectrum of communication drivers [Rus11].

C.3 **Execution services to partitions: application Programming Interface** The execution services to partition are provided by the guestOS and the hypervisor.

The development of MCS can be enabled by means of strict space and time partitioning such as that proposed by the ARINC-653 specification [ARI05], which defines an API called APplication EXecutive (APEX) that allows multiple applications with different safety levels to be executed in the same hardware platform (core module in the context of ARINC-653). One promising approach to build partitioned systems is that based on a hypervisor [HJ14], a minimal layer of software with a low overhead that supports mixed-criticality partitions built on top of operating systems with different purposes. However, this API lacks of a more general view to deal with partitioned systems. In DREAMS, a API has been defined (DRAL) to deal with that services. A standarization of these services should be appropriated for the new developments.

State-of-the-art

S.1 ARINC653 [ARI05] Standard is a set of documents that define the interface of partitioned systems. ARINC-653-P1 document defines a general purpose execution service, ARINC-653-P2 extends that services.

S.2 ARINC-653-P4 is the document that defines a subset of ARINC-653 services to increase the certifiability of the execution layer. This subset removes the process services and permit only one periodic and aperiodic process per partition. Internal communication services such as Semaphores, Buffers, Blackboards and Events are also removed. The defined services are closed to the DRAL services defined in DREAMS

C.4 **Timing interference mitigation mechanisms**

One of the main problem in multicore systems is the timing interference. It is produced by the use of shared resources such as cache, memory, buses, etc. It generates unpredictability on the execution of the application activities. There is a need of control of the shared resources and reduce the impact of the non critical activities on the critical ones.

State-of-the-art

S.1 Several works have pointed out the problems of shared resources in multi-core systems. In [ABD+13], it is reviewed the impact of shared buses, caches, and other resources on performance and performance prediction. In [CCP+15], a bus protocol based on TDMA-based memory arbiter jointly with a second, dynamic arbitration layer facilitates the interference-free integration of mixed-criticality applications. In [GSHT13] a global time-triggered scheduling approach with barrier synchronization is proposed for multicores.

S.2 In [GJR+15], a control of the running tasks accessing to shared resources is presented. a control oriented deterministic platform software is presented. In [YYP+13b], it is defined a memory guard mechanism that regulates the memory accesses. It is a regulation oriented mechanism that allocates a maximum bandwidth usage per timeslots. In [KRF+14], it is proposed a distributed run-time WCET controller that stops low-criticality tasks (running on other cores) whenever it determines that their continued execution could cause a high-criticality task to fail to meet a deadline. In [CSB+17], it is proposed a controller scheme for critical cores that is transparent to the applications and is implemented at hypervisor level (XtratuM). It defines the mechanisms that are set by the hypervisor when slots of critical partitions are executed. It uses the monitor performance unit of the processor.

C.5 **Virtualization Layer performance analysis and characterization** Virtualization layers can have multiple factors that affect to their performance. Many challenges in performance analysis of virtual environments are due to hypervisor implementation, guest operating system, guest resource interdependence, and utilization ambiguity. Application or guestOS uses the hypervisor services to deal with the hardware. Several traditional OS services are affected by the virtualization layer such as context switches, latency to interrupts, time management, etc. There is not a general and consolidated metric to analyze the impact of the virtualization layer. In RTOS there are several metrics that could be partially used, however, they should be extended and adapted to deal with the virtualization view.

State-of-the-art

S.1 There exists consolidated metrics and benchmarks for RTOS such as Rhealstone metric that provides a set of parameters to compare different RTOS. However, virtualization layers does not achieve the maturity to consolidate a set of indicators to compare different implementations due to the wide range of solutions.

S.2 Several works [RW14] [HHL+97] [VMW] [**?**] [CCP+15] analyze the performances of the virtualization layer considering shared resources or different implementations (L4 microkernel, VWare, XtratuM).

S.3 Hybrid solutions for virtualization combining para-virtualization and full virtualization should improve the performances of the virtualization layer.

C.6 **Consolidation of real-time applications along with General Purpose OS** An important challenge in the design of embedded systems is the consolidation of software applications with different levels of criticality on a common hardware platform. A common practice to isolate safety critical applications is through the proliferation of multiple hardware platforms, which are dedicated to basic operations. This is a highly inefficient way of using the available processing power since many of these platforms are typically not used at their full potential. However, recent multi-core architectures with new hardware extensions (e.g., virtualization, TrustZone) enable the execution of multiple applications on the same platform safely and securely, thus reducing costs and weight, helping to increase efficiency. The consolidation of different OSes on the same platform implies the concurrent execution of a critical OS with stringent real-time requirements with non-critical applications. In this context, the main challenges are to integrate real-time tasks execution with software applications of a GPOS.

In the past, virtualization has been presented as a solution to isolate OSes in a VM. This approach offers the advantages of reducing implementation costs by abstracting the host platform. Additionally, features provided by hypervisors, such as memory partitioning, CPU and interrupts abstraction help the OSes isolation. However, the use of a hypervisor may cause performance overheads, and therefore the critical execution path of real-time operations may not be ensured.

In this context, a secure monitor layer, (known as VOSYSmonitor which is its product name), has been developed in the DREAMS project. This software compononent enables the native concurrent execution of a safety critical RTOS (or another type of OS) along with a GPOS with the option to use virtualization extensions, such as Linux/KVM, in order to instantiate a variety of different VMs. It guarantees the isolation of critical real-time tasks, while minimizing the overhead on the global execution. The monitor layer is the highest secure operating mode available on ARM processors, designed with the hardware security extension ARM TrustZone, which manages the interaction between two execution worlds. In this context, VOSYSmonitor has been designed for the ARMv8-A architecture by guaranteeing peripherals and memory isolation between both OSes with ARM TrustZone. Moreover, this software layer has been designed to meet certification requirements induced by mixed-criticality systems. The main advantage of such a solution is to allow dynamically cores sharing between both applications, thus offering a close to native performance. To achieve this, VOSYSmonitor supports a context switch mechanism with a minimal overhead.

State-of-the-art

S.1 Among existing solutions, TrustZone-assisted hypervisor[PPG+16] is able to run an arbitrary number of RTOSes in virtual machines. In this design, only one RTOS is executed at a time in the Normal world, while the context of the other guests is preserved in the Secure world. Therefore, if another RTOS needs to be scheduled, the current RTOS execution is stopped and its context is saved in the Secure world, then, the TrustZone-assisted hypervisor restores the second RTOS context in the Normal world. Real-time requirements of inactive RTOSes are met by setting their interrupts as secure, thus allowing to preempt the running RTOS, which uses normal interrupts. Such a implementation does not allow an efficient cache management since all RTOSes are scheduled in the Normal world, which requires to clean/invalidate cache memory during context switches, thus increasing the overhead. Moreover, at the time of writing, the TrustZone-assisted hypervisor only supports single-core configuration. VOSYSmonitor is able to run on multi-core heterogeneous platform and take benefit of cache memory to reduce the context switch overhead. Indeed, since OSes are assigned to a specific world, VOSYSmonitor can rely on the isolation of caches lines provided by ARM TrustZone, thus avoiding the need of cache operations during a context switch.

S.2 Other solutions enable the co-execution of two or more OSes using ARM virtualization extensions. A design based on Erika OS[Ava15] is executed along with Linux on top of XEN hypervisor[FOu]. This solution has been implemented on a cubieboard2[aar] by assigning each OS on a different core. While it ensures the isolation, there is a risk of an inefficient use of computing power if Erika OS has a low workload. To prevent such a case, VOSYSmonitor is able to reallocate core resources to the GPOS if the RTOS has no real-time tasks to schedule. Others solutions based on virtualization extensions are Xtratum[MRCM09] and NOVA[SK10]. Furthermore, the isolation of Virtual Machines (VMs) against the host is not guaranteed. Indeed, some breach in the hypervisors can allow VMs to cause a denial service or access data from the whole system. As a matter of fact, a security vulnerability named VENOM[Gef], allows an attacker to escape from the isolation of a VM and get the access to the host and the others VMs.

S.3 ARM Trusted Firmware[Ltd16] hereafter referenced as ATF is a software layer able to host a Trusted Execution Environment alongside a Non-Trusted software. At the best of our knowledge, the current implementation of ATF only supports a dispatcher to execute OP-TEE OS[Lin] in the Secure world in order to provide trusted services to the Normal world application. VOSYSmonitor overcomes this limitation by giving the possibility to concurrently execute an RTOS and a rich OS.

S.4 A different approach, such as FLexPRET[ZBSL14], introduces a new multi-threaded processor intended for mixed-criticality systems where threads are classified in two categories: *hard real-time thread (HRTT)* where deadlines must be met and *soft real-time thread (SRTT)* where a time constraint non-respected is acceptable. Another solution is IDAMC[MDB+12], which proposes a platform where multiples nodes are interconnected by a network-on-chip. These nodes includes processors based on the LEON3 technology connected through routers monitoring access to shared resources. While these solutions have been created for mixed-criticality systems, the main disadvantage is related to the new processor or cluster architecture, which implies, for instance, a higher workload for developers and increases the risk of unknown hardware/software bugs since the architecture is new and not widely used. On the other hand, VOSYSmonitor is based on ARMv8-A, one of the most popular embedded architectures for mobile, automotive, drone markets.

C.7 Certifiable virtualization solutions for mixed-criticality systems

In mixed-criticality domains, the term functional safety has become a topic of high importance. Indeed, functional safety generally means that malfunctions of the operating system, which contain mission-critical tasks, that lead to any kind of threat or even accident have to be avoided or mitigated. Therefore, it is fundamental in the field of functional safety to identify and understand potential risks and failure causes of a system. If ideally all potential failure causes are known and the consequences understood, it is possible to define countermeasures. Thus, failures are detected before a hazardous event occurs and with the needed functional safety reaction the safe state is initiated. The safe states can importantly vary according to the final application as well as the injuries, which might be led by the system failure without countermeasures. As every application is different and has its own particularities and thus potential failure causes and related safe states, the functional safety analysis is very interesting challenges. In this context, many functional safety standards have been established to define the main requirements to fulfil during the development of critical systems in order to ensure a high level of reliability in the critical systems. The main functional safety standard is the IEC/EN 61508 that defines the basis for functional safety developments for E/E/EP (electronics, electronic or programmable electronic) applications. In addition, the IEC/EN 61508 is expanded by additional industry sector specific standards, such as the ISO 26262 Road vehicles Functional Safety which has been specially defined for the automotive domain.

In the scope of DREAMS, the main strategy has been based on the development of a first VOSYSmonitor solution for automotive sector and critical systems must be compliant with the ISO 26262 standard. Therefore VOSYSmonitor has to be certifiable with the same Automotive Safety Integrity Level (ASIL) than the critical system running on top of it.

Also in the scope of DREAMS, hypervisor based on XtratuM safety cases have been defined. A safety case is a structured argument supported by a body of evidence that provides a compelling, comprehensive and valid case that a system is safe for a given application in a given operating environment. The DREAMS approach to certification provides a number of safety cases for foundational components (e.g., hypervisor, networking, etc.). When these components are integrated in a MCS, then a specific global safety argument for this MCS can be also assembled to show the validity of the safety claims.

State-of-the-art No similar solution for ARMv8-A architecture has been identified. Only Sierraware proposes a solution to extend the TrustZone monitor on the ARMv7-A architecture, which is not compatible with the AArch64 mode of the ARMv8-Architecture. Another demonstration, performed by Freescale, is able to run concurrently, on the i.MX6 target, non-critical systems along with Autosar OS which is isolated by means of ARM TrustZone. However, the i.MX 6 hardware platform is composed by Cortex-A9 processors (e.g., ARMv7-A architecture) and, again, it is not compatible with the AArch64 mode of the ARMv8-Architecture.

However, some alternatives, related to consolidation on a common hardware, based on a hypervisor solution, are already certifiable and available on the market. The main commercial products are:

S.1 SYSGO - PikeOS Hypervisor: PikeOS has been designed for use in safety-critical applications according to several functional safety standards. The PikeOS micro-kernel consists of less than 10.000 lines. Thanks to strict separation technology, applications of different criticality levels, real-time or non-real-time can run concurrently in a mixed critical environment on a single standard hardware platform (ARM platforms supported: Renesas R-Car H2, R-Car H3 planned, Xilinx Zynq, NXP i.MX 6).

S.2 GreenHills - INTEGRITY multivisor: INTEGRITY Multivisor can host arbitrary guest operating systems alongside a comprehensive suite of real-time applications and middleware. Products in the GreenHills Automotive Platforms have achieved the highest levels of safety certifications including ISO 26262 - ASIL D (ARM platform supported: Freescale i.MX and Layerscape, Nvidia Tegra, Renesas RCAR, Texas Instruments Jacinto, OMAP, and Sitara).

S.3 QNX OS - QNX: As a true microkernel OS, it provides inherent protection and isolation for safety-critical software components regardless of whether the system comprises of only safety-related components or a mix of safety and non-safety components. The QNX OS has been certified for automotive safety according to the ISO 26262 - ASIL D.

S.4 XtratuM: As a hypervisor explicitly designed for embedded real time systems, it provides protection and isolation for safety-critical software components and multicore systems. It is in process of certification based on ECCS standard that is used in space applications. Initial XtratuM models for multicore systems have been validated in Multipartes and DREAMS projects for safety case certification [LPO15a] [**?**].

10.1.1 Problems Not Solved in DREAMS

I.1 **Time interference guarantees**
Shared resources in multicore systems introduce time interferences in the execution of applications. Critical application require to analyse the WCET of the internal tasks and have guarantees of the resource allocation. This is a complex issue that involves the modelization of the activities, theoretical foundations for the system analysis, virtualization and guestOS mechanisms, execution services and hardware support. The use of performance counters at hardware level is use in several approaches to control the execution of partitions [KRF+14] [YYP+13b] [CSB+17].

I.2 **Integration of legacy code and OS**

On a para-virtualized environment, guestOS require to be adapted to be executed on top of the hypervisor. Legacy applications can be executed on top of the guestOS without modifications. However, if the application require some kind of communication or access to other application running on the virtualized environment, they need to use the virtualization services and , in consequence, to be modified. In DREAMS project, XtratuM hypervisor has been redesigned to use hybrid virtualization allowing the execution of legacy applications and OSs on top of it. In the WindPower demonstrator, WindowsCE has been executed allowing legacy Windows application to run on the virtualized environment.

I.3 **Integration of static and dynamic scheduling policies**

XtratuM uses a static scheduling policy to execute the application partitions. Partition priorities can be used also as criteria for scheduling allowing a dynamic execution of partitions. However, these dynamic policies should be analysed and experimented in real applications to evaluate the priority assignation and latencies obtained. Other issue is how to integrate in the configuration file the information related to the partitions.

10.2 Innovation for Road-map

I.1 **Integration of hardware support virtualization**

Specification of the functions, services and boundaries of the virtualization layer when para-virtualization and full virtualization are used.

I.2 **Extending hypervisor mechanisms to support NoC and TTEthernet devices**

Specification of the virtualization requirements and services to support communication services at hypervisor level.

I.3 **Definition of the platform independent DRAL services**

Para-virtualization services deal with the services that a partition application can use in order to get the status of the system or other applications, control the execution of others and communicate with them even if other partitions are allocated in other cluster of a distributed system. DRAL services can be considered as a starting point to discuss the service standarization.

I.4 **Define a scheme for reconfiguration**

System reconfiguration has been experimented in DREAMS to manage fault situations. Although it has been successfully achieved, a scheme for a more general approach combining applications and hypervisor should be considered.

11 Certification, Safety and Dependability

11.1 Research Challenges

Our society increasingly depends on a broad variety of computer-controlled systems (safety-critical and dependable embedded systems) where failures are critical and may have severe consequences on property, environment, or even human life. For example, the braking control of the car, railway signalling systems for trains and energy generation and distribution protections.

Those safety-critical embedded systems are commonly certified according to international safety standards. Certification is the process that demonstrates the compliance of a product, system, subsystem or element, with established standards. Safety certification guarantees that a product, system, subsystem or element is safe enough for its purpose with a given confidence level [LPO15b].

As systems become more and more interconnected (e.g., Internet Of Things (IoT)), security and cybersecurity become a key challenge for the design, development, deployment and maintenance of safety critical embedded systems. In the future, we should consider that there will be no safety without security. The relationship between the terms security, safety and dependability are analysed in [ALRL04].

Safety and safety certification research challenges should address industrial functional safety requirements such as the ones described in the previous sections (Sections 2, 3 and 4), considering both the natural evolution of these domains and new disruptive technologies [Per17, mcs12].

- **Natural evolution:** In multiple domains the embedded systems architecture has followed a federated approach, in which the system is composed of multiple interconnected subsystems where each of them provides a well-defined functionality. The ever-increasing demand for additional functionalities has lead to an increase in the number of subsystems and networks that limit the future scalability of this approach (see Sections 2, 3 and 4). In addition to this, the overall system reliability can be decreased due to electrical failures caused by an increasing number of connectors and wires [SM99, SG01, PGN$^+$14b, PGN$^+$14a]. Therefore the adoption of integrated architectures that enable the integration of functions of different criticality (a.k.a. mixed-criticality) becomes relevant for the future natural evolution of such systems, supporting the reduction of subsystems, networks, cables and connectors. Moreover, this potentially leads to a decrease of the cost, size, weight, power and the increase of the overall reliability.
- **Disruptive technologies:** The deployment of disruptive technologies such as Autonomous Vehicles (AUV), Industry 4.0, IoT and Artificial Intelligence (AI) will further increase the need for higher computation and connectivity capabilities while ensuring that deployed solutions are cost competitive, dependable, safe, secure, reliable and meet application constraints (e.g., size, weight, power). All in all, integrated architectures that enable the integration of functions of different criticality (a.k.a. mixed-criticality) also becomes relevant and necessary.

C.1 **Safety certification for integrated architectures based on modern multi-core devices**

The integration of applications of different levels of criticality in a multi-core device leads to several challenges concerning safety certification standards.

- **Temporal and spatial independence:**

Multi-core mixed-criticality systems are prompt to spatial and temporal interferences. Spatial interference means that the data used by one element is modified by another element, which should not occur in an IEC 61508 compliant item. On the other hand, temporal interference means that one element causes the incorrect function of another element. For example, taking too high a share of the process execution time or by locking a shared resource, which blocks the enforcement of another element.

IEC 61508 safety standard defines the resource sharing, processor time sharing, peripheral sharing, the communication between the components to achieve the overall system design and the fault-propagation as the common causes of interferences of computing systems. For example, multi-core devices such as Zynq-7000 and P4080 uses shared memories, coherency management units and direct memory access to communicate, manage coherency of the caches and core processors and to manage the access to the memory. Those components should be diagnosed to guarantee that they do not impact on the overall system execution. E.g., injecting faults which could affect the temporal behaviour or modifying the data of the memories.

Due to their complexity the use of the above elements in safety applications also require fault avoidance measures to be followed. Fault avoidance measures for the use of these components are not yet part of IEC 61508 and require further research while some hints were provided within the DREAMS project with Document D5.3.1.

- **Diagnosis:**

  Microcontrollers, which are used in safety-related applications according to IEC 61508, must fulfil the Safe Failure Fraction (SFF) depending on the targeted SIL and Hardware Fault Tolerance (HFT). This requirement cannot be fulfilled without any diagnostic technique, implemented either in software routines or by hardware blocks.

  Not just the microcontroller as a complete component has to fulfil this requirement, but also each element inside of it. Examples of decomposed elements of a typical microcontroller are timers, DMA, registers, internal RAM, code execution block, flag register, program counter, stack register and analogue/digital converters.

  Annex A of IEC 61508-2 provides diagnostic techniques together with their respective maximum achievable diagnostic coverage. As a result of this test routines, for those elements mentioned above, are not really a challenge for a software designer. This situation changes dramatically in case the designer has to use multi-core processors, e.g., due to performance reasons. For example, each core has its own local caches, but all of them are sharing, for example, the same program and data memory.

  It is evident that further elements are necessary, which need to organize and arbitrate the accesses to and from those called shared resources. Besides the arbitration and use of share resources, the coherency of the local caches and global interrupt handling are further important topics, which need to be considered in multi-core processors. For example, the element which is providing the cache coherency in ARM microcontroller architectures is called Snoop-Control-Unit (SCU).

  A more general term for this element is the cache coherency unit. Their ability to manage the accesses to shared resources is essential for the high performance of modern multi-core processors. The basis for the high-performance arises on the algorithm inside of those units, which is implemented by the manufacturer. The better these algorithms are the higher the performance and the higher the success on the market will be. Algorithms and structures of those units are top secret and their complexity can exceed the complexity of the microcontrollers themselves.

From the functional safety point of view, those elements need to be diagnosed to provide sufficient measures for fault control. The traditional diagnostic techniques, as stated in the standard, can basically be implemented, but their diagnostic coverage is questionable due to the complexity of those elements. This requirement leads to the following questions:

(a) Which kind of diagnostics shall be implemented to provide a required SFF? What is the diagnostic coverage and how can the claimed diagnostic coverage be proven without any support provided by the manufacturer?

(b) Instead of direct, traditional diagnostic measures for these elements, more generic, application specific methods could be a matter for future research.

(c) Similar to the generic diagnostic measure coded-processing, where the element is nor directly in the scope, but transforming the data and processing twice in a different manner with final re-transformation and last data compare.

(d) How can the effectiveness of such generic diagnostic measures be proven? These questions are not answered yet, and appropriate diagnostic techniques are missing.

- **Device fault avoidance of systematic failures:**

  Besides the diagnostics for fault control of random hardware failure, the IEC 61508 request to consider measures for fault avoidance of systematic failures during the development. This also applies to multi-core processors. In case of a mass-product this is implicitly given, according to 7.4.6.1, IEC 61508-2: This is because the likelihood of faults in such devices is minimised by stringent development procedures, rigorous testing and extensive experience of use with significant feedback from users. Otherwise, the devices have to be considered as a new device or ASIC.

State-of-the-art As stated in [PGN+14a], multiple reports [Inf, eas12, EAS13, EAS11, BR10, ubm13], research projects [TOG+14, CER13, MUL14, SAF17a, CON14, EMC14, SAF17b, DRE13] and publications [KOESH07, MEA+10, Ern10, PGN+14a, PGN+14b, PGN+14a, Per17] indicate that is likely to be a significant increase in the use of multicore processors over the next years replacing applications that have traditionally used single core processors. Multicore and virtualization technology, independently or in combination, can support the development of integrated architectures in mixed-criticality platforms by means of software partition, or partition for short.

IEC 61508 safety standard does not directly support nor restrict the certification of mixed-criticality systems. Whenever a system integrates safety functions of different criticality, sufficient independence of implementation must be shown among these functions [IEC10a, IEC10b, PGN+14a]. If there is not sufficient evidence, all integrated functions will need to meet the highest integrity level. Sufficient independence of implementation is established showing that the probability of a dependent failure between the higher and lower integrity parts is sufficiently low in comparison with the highest safety integrity level [IEC10b]. As stated in IEC 61508-2 Annex F, spatial and temporal independence are key to ensure the independence of execution ("that elements will not adversely interfere with each others execution behaviour such that a dangerous failure would occur").

Widely available COTS multicore processors were (and are) not designed with a focus on hard-real-time applications but towards the maximal average performance. This is the source for multiple temporal isolation/independence issues [KNP+13, NST+13, PGN+14a, PGN+14b] and a challenge for the development of safety critical embedded systems as stated by different experts in the field [SBR10, Fuc10, SEH+12, ACQ+11]. In general, providing sufficient evidence of isolation/independence among safety and non-safety related functions distributed in a multicore processor is not a trivial task [KNP+13, NST+13, PGN+14a, PGN+14b].

C.2 **Safety and security in integrated architectures (mixed-criticality)**

In integrated mixed-criticality architecture, the fact of spending a lot of effort in guaranteeing that the system is safe is a common practice. The main reason of that is that those systems generally need to be certified. Safety-related standards define concise safety processes that shall be followed to achieve a certified system.

Ultimately, integrated architecture designers have begun considering security as a promising working line. Indeed malicious attacks on the system may cause failures and catastrophic events. So, there is a need to assess the safety and security properties of critical embedded systems to create dependable system architectures. The combination of those disciplines in integrated architectures is the focal point for nowadays integrated architecture developers.

State-of-the-art

S.1 Safety and security [BO16]

C.3 **Digitalization of systems (e.g., Industrial Internet Of Things (IoT), Industry 4.0)** The trend towards digitalization of systems such as wind-turbines needs to combine the dependability properties of the system such as safety with security / cybersecurity challenges associated to the interconnection of systems to the industrial IoT.

State-of-the-art

S.1 Internet of Things for Industrial Automation [KJE+17, BS, TMKPF, BO16, BD16]

C.4 **Safety standards and integrated architectures (mixed-criticality)**

Safety domain standards such as IEC 61508 define "techniques for achieving non-interference between software elements on single computer" (see Annex F of IEC 61508). However, those methods are not applicable at all to multi-core integrated architectures where one resource is shared by the processor's components. For example, multi-core devices typically implement a L2 cache (shared memory) to communicate the processing cores, which commonly leads to spatial and temporal interference.

State-of-the-art

S.1 [LMB+17, PGN+14b, PGN+14a] defines solutions applicable to partitioned and networked multi-core mixed-criticality systems.

C.5 **Cost-effective certification of safety product families in integrated architectures (mixed-criticality)**

Product line technologies build upon the re-usability of the underlying software technologies (e.g., objects, components, services) form a promising approach to reduce the cost of certification and recertification, by maximizing reuse of certified units, hereafter called "pieces". Reuse is always challenging as it is difficult to ensure that third-party components will perform correctly in an environment for which they were not explicitly designed. Among others, the $500 million crash of Ariane 5 in 1996, remains a strong evidence of the opposition between reuse and verification and in turn, certification.

State-of-the-art

S.1 [NEL+17, LMO+] present a generic and modular product-line safety argumentation scheme.

11.2 Innovation for Road-map

I.1 **Generic IEC 61508 compliant modular safety cases**

Within DREAMS research project four modular safety cases that may be used as guidelines for future developments are generated. Those MSCs define the safety arguments that a safety hypervisor, partition, COTS multi-core device and mixed-criticality network must fulfil to be compliant with IEC 61508 [Lar17, LPO15b, LPO15c].

On the other hand, several linking analyses where the way in which the safety arguments stated in the MSCs are met by some custom and commercial components (e.g., XtratuM hypervisor, Zynq 7000 multi-core device, TTEthernet network) are defined.

I.2 **Multicore usage constraints for safety applications**

The use of multicore specific elements requires the research and definition of suitable generic diagnostic and fault avoidance measures compliant with appropriate certification standards.

I.3 **Mixed-criticality cross-domain patterns**

Cross-domain patterns are used in this project to guide and support engineers to solve the commonly occurring issues related to mixed-criticality systems. Those patterns are used in this research project to define several solutions to the commonly occurring problems of mixed-criticality systems based on partitioning, networking and multi-core processors [LAO+16, LMB+17].

I.4 **Generic and modular product family safety argumentation scheme**

A generic an reusable safety argumentation scheme for a product line is generated under DREAMS research project [LAO+16, NEL+17]. This argumentation scheme contains four abstraction layers (product line, product sample, component, MSCs) where the safety-related arguments and evidences required to attain certification are included.

On the other hand, this argumentation scheme may vary from standard (such as safety, security, real-time) and requirements. Those variation points allow future adaptations of a product line to other domain-specific standard or modifications to the requirements.

I.5 **Safety certification of systems based on multicore / manycore and visualization solutions**

As stated in the introduction, the natural evolution of industrial systems such as avionics, automotive and wind-turbines (see Sections 2, 3, 4) must cope with the continuous integration of additional functionalities, connectivity and intelligence. The federated approach strategy is reaching scalability, reliability and power-weight-volume limitations to be addressed. The development of mixed-criticality integrated architectures based on multicore and partitioning is a suitable solution that needs to be further addressed in order to increase the competitiveness of associated industries [Per17].

In addition to this, the development of safety products based on new disruptive technologies such as AUV, Industry 4.0, IoT and AI will face equivalent limitations where mixed-criticality is a suitable solution to enable their development and increase the competitiveness of associated industries [Per17].

This innovation also requires the update of current safety standards (e.g., IEC 61508) in order to clarify, guide and standardize current state-of-the-art with respect to the usage of multicore devices and partition solutions for the development and certification of mixed-criticality systems.

I.6 'No safety without security'

The adoption of new disruptive technologies such as Industry 4.0 (e.g., digitalization) and IoT leads to several challenges with respect to safety systems, e.g., Industrial security, cybersecurity. As security / cybersecurity could jeopardize the safe operation of the system, associated industries need to face this challenge in order to protect society while ensuring competitiveness of associated products. This is an innovation challenge for the natural evolution of systems and development of safe products based on new disruptive technologies [Per17].

12 Model-Driven Engineering

DREAMS model-driven engineering process. The DREAMS architectural style and the reference platform consisting of networked, virtualized multi-core computers that are controlled by hierarchical resource managers tackle the MCS integration problem at the platform level [DRE14]. The Model-Driven Engineering (MDE) process and the tool chain developed in the project [BDM+17a] enable to deploy mixed-critical applications onto the DREAMS platform and constitute its interface to system designers and integrators. The design of the architecture and the underlying assumptions had the following impacts onto the DREAMS MDE process:

- Use of design-time configurations that control the access to shared resources to ensure the freedom of interference of mixed-critical applications (as mandated by safety standards) by means of temporal and spatial segregation.
- The approach considers distributed real-time system with a bounded fault-hypothesis under a closed world assumption that uses adaptation to enable fail-operational behavior in scenarios considered at design-time (i.e., reconfiguration and degradation in the event hardware faults).

Illustrative scenario. To make the discussion of research challenges more tangible, we exemplarily discuss adaptation scenarios of autonomous cars[1]. Autonomous cars are highly complex HW/SW systems that have to interact with a changing system context which is defined by physical quantities that can be measured by sensors (e.g., traffic situation, light conditions, etc.), or by the presence of interacting systems (i.e., other autonomous cars, or elements of a “smart” infrastructure).

Autonomous cars have to adapt to the current system context defined by the physical environment instead of providing a static system that covers any possible driving situation (e.g., traffic jam in urban

[1]These challenges also exist for MCS in other transportation domains, smart factories or the internet-of-things, etc.

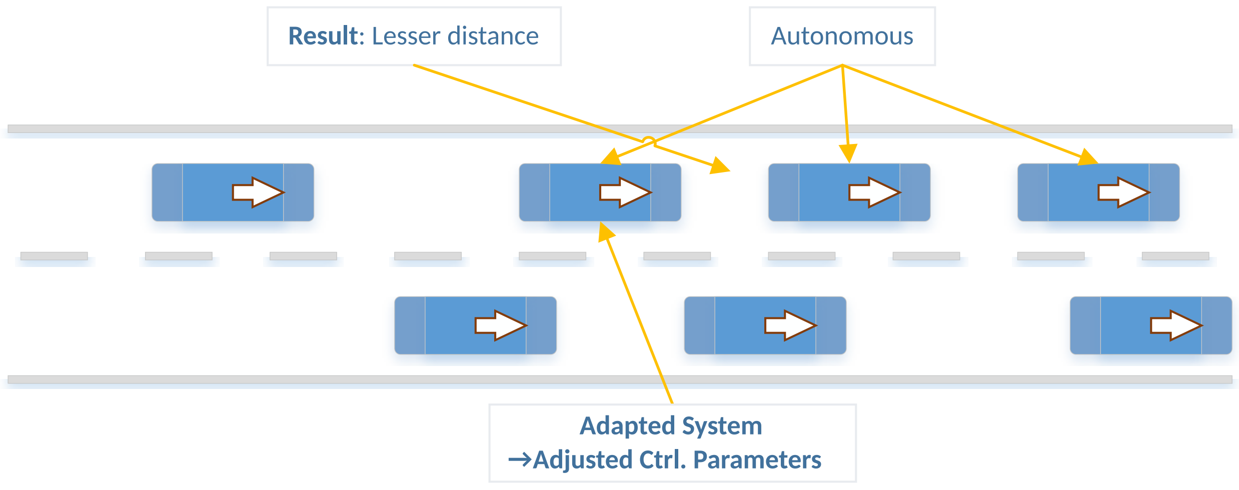

Figure 12.1: Platoon of autonomous cars on a freeway: Adjustment of distance control.

area, driving a freeway at night). Instead, the system monitors itself and the environmental context to adapt itself by tuning parameters of algorithms, or by switching to a dedicated system mode.

The interaction and collaboration with other systems has a similar impact and leads to an open-world assumption that also has to be considered for the development process. A typical example is a group of autonomous cars that is driving along a freeway. These cars can form a platoon to optimize their operation in a system-of-systems (see Figure 12.1) where, for instance, the speed and distance control algorithms may be adjusted to decrease the inter-car distance, relying on the availability of sensor data and control commands provided by cars that run ahead.

12.1 Research Challenges

In this section, we will characterize challenges for the development process of future MCS.

C.1 **MDE for adaptive and collaborative mixed-critical systems.** The development process presented in DREAMS yields system configurations that do not take into account adaptations to unforeseen changes in the system's environment, including the structure and composition of the system itself. To lift this restriction, we proposed to apply *MDE for adaptive and collaborative mixed-critical systems* such as the autonomous car example introduced above.

Since it is impossible under an open-world assumption to consider all possible scenarios at design-time, capabilities of the run-time infrastructure (e.g., plug-and-play middleware architectures) as well as interface and service models of the constituent systems must be explicitly described and taken into account. This raises the following challenges:

First, the uncertainty about the environment and the run-time adaptation capabilities of the system may result into a huge design-space regarding the system's parametrization, or the combination of operating modes. This additional complexity will complicate the certification of adaptive MCS, for which additionally the fault-hypothesis and the corresponding measures to protect critical functions have to be considered.

Second, collaborative systems form a system-of-systems (SoS) that needs to be described such that assumptions about the run-time behavior can be made at design-time and guarantees can be given (e.g., an input to a safety-critical software component may stem from a component running in another system and therefore has to be certified according to the same safety integrity level). Suitable descriptions may include metamodels that enable to reason about the properties of the SoS at design-time, but that would also be used by the run-time software to verify the assumptions that have to be met in order to establish a link with another constituent systems.

In addition to SoS models and the verification methods discussed so far, further parts of the system development process can be linked with corresponding run-time counter-parts. Most importantly, adaptivity and system collaboration shift parts of the design-space exploration phase to run-time (e.g., software component-to-platform deployment and component-to-component binding). This approach allows to shift functionality between the constituent systems of an SoS, select different fidelity levels of the systems functionality, or to alter its configuration. E.g., in the autonomous car scenario, sensing functionality could be shifted to the foremost car that provides the corresponding signals to all cars in the platoon.

State-of-the-art Salehie and Tahvildari survey self-adaptive software with a focus on system properties and the required run-time mechanisms, but the authors also compare engineering approaches for adaptive systems [ST09]. Lemos *et al.* present a roadmap [dLGM+13] and overview for MDE in adaptive systems [BHRV13, RGSZ14] that discusses various modeling aspects and categorizes the corresponding design-spaces. Further, Lemos *et al.* point out that typically an engineering process for adaptive systems is situated between the static processes for traditional systems, and pure online approaches [dLGM$^+$13], balancing the required dependability guarantees and the flexibility [IT09] (however, without explicitly considering criticality and certification aspects which are crucial for MCS). As a consequence, the usual order of the design activities (requirements analysis, design, implementation, etc.) and the traditional time-line of the system life-cycle phases (development time, deployment time, run-time. . .) are broken up, which requires the availability of system models also at run-time [MBJ$^+$09]. Happe *et al.* discuss the role of models in self-adaptive and self-healing systems that dynamically react on changes in the environment and distinguish between design-time models (used to optimize or analyze the system prior to its execution), and run-time models used during the execution of the system (e.g., to provide adaptive behavior) [HKB$^+$09].

Cheng *et al.* categorize modeling techniques for adaptive systems [CdLG$^+$09] and define the corresponding modeling viewpoints [ALMW09], e.g., to describe the goals of adaptive systems, (environmental) change as the cause for adaptation, or the mechanisms used to react on changes. Sama *et al.* investigate requirements engineering for adaptive systems which must cope with their inherent uncertainty, and which must be considered hand in hand with the system's goals to provide context-awareness (based on self-reflection and validation techniques) [SRWE08].

Another approach to adaptive MDE expresses adaptiveness using so-called changeable (logical) components and also considers a mapping of such systems to a middleware platform [RMMG08], but misses an overall concept to handle adaptiveness across different design steps. Other relevant aspects of MDE for adaptive systems include the consideration of adaptiveness as variability [BGF$^+$08], the augmentation of run-time artifacts with model informations [MBJ$^+$09], and the consideration of cloud services [FRC$^+$13].

Collaboration of distributed embedded systems is one of the key aspects of the internet-of-things (IoT) (c.f. for instance the INTER-IoT project [INT] that develops a framework to foster the interoperability of different IoT platforms). A number of different MDE approaches have already been applied to this area [CS17], but which do not consider certification and other aspects relevant for MCS. In this context, Ciccozzi *et al.* discuss an approach to tackle criticality issues for IoT [CCR$^+$17], whereas projects TAPPS [TAP] and PASS [PAS] propose an eco-system including a platform, and app-store and a tool-chain ensuring the safe and secure integration of apps developed by third parties.

C.2 **Collaborative iterative engineering of MCS starting from underspecified models.** To make system development more efficient, (e.g., to shorten the time-to-market and to reduce certification cost), we discuss research challenges in the area of *collaborative iterative engineering of MCS*. Our proposal promotes to use possibly underspecified models that are collaboratively edited by engineers from different system domains who simultaneously work on artifacts from different design phases (e.g., requirements specification, design models, etc.). The approach entails to continuously verify design constraints, in order to parallelize the development process and to front-load feedback onto the design into early phases.

Current MDE processes for (mixed-)critical systems are typically based on the V-model and employ metamodels to describe the artifacts of the different phases. Based on metamodels for requirements, architecture design, functional and component design, these approaches allow for the step-wise refinement of a requirements specification into a design or an implementation model (from which deployable artifacts could be generated automatically). Based on the underlying metamodels, tools can be used to automate the transformation steps between the different phases, and also the corresponding validation steps (e.g., automated testing or formal verification at the component or design level). However, this method suffers from the problem that some errors in the design can only be detected in late phases of the design, which requires managing costly change requests to correct the original problem. Furthermore, the process can only be parallelized within the currently active phase (e.g., simultaneous design of hardware platform and design of application software), since the artifacts from the preceding phases have be to available (e.g., complete requirements specification).

In order to speed up the development, future MDE processes for MCS should allow to use underspecified artifacts as a preliminary basis for later phases of the development process. E.g., this would allow to start working on the system design while the requirements model has not been finalized yet. Since also an incomplete requirements model defines constraints, validation can be performed immediately while the design model is being created. This approach allows to efficiently explore "spike solutions", and exploits all information about the system that has been modeled so far. A major prerequisite is the availability of tools that not only automatically derive and update refined artifacts (e.g., based on design rules), but also perform continuous validation and verification of the still underspecified system. This front-loading of validation and verification activities provides immediate feedback to the system engineers and helps to decrease the round-trip time during the design phase.

In case a violation of a constraint or requirement is detected, possible modifications to remedy the problem should be proposed automatically. On the one hand, the immediate feedback enables to revert erroneous modifications. On the other hand, design-space exploration should for instance propose modifications to a software architecture when the corresponding platform model is changed (e.g., use of a different safety architecture in case a processor is added or removed).

Under an open-world assumption (c.f. above discussion on *MDE for adaptive and collaborative mixed-critical systems*), this approach could be taken to the extreme and parts of the design-space exploration steps such as deployment and binding could be shifted to the run-time phase, together with the corresponding verification steps. However, as Zeller and Prehofer point out, this approach might put certification at risk, in case highly complex solvers are used in the online exploration process [ZP13].

State-of-the-art The iterative MDE discussed above has commonalities with agile development methodologies for which MDE approaches have been discussed [Mat11, WHR14]. Di Ruscio *et al.* describe a co-evolutionary approach to handle the refinement of models [DRIP11], with a focus on the generation of artifacts in subsequent phases of the design process. Another relevant approach is to consider refinements as model transformations [PG08], especially for the deduction of constraints for refined elements.

Model verification and constraint modeling techniques developed for sequential development processes constitute an important starting point for the proposed iterative MDE approach. OCL is part of the UML standard and can be used to describe design constraints, e.g., in terms of invariants or pre- and post-conditions [OMG12]. This also applies for contracts, a technique that is used to support the distributed collaborative design of complex large-scale embedded systems in a modular fashion [BCN$^+$15a, BCN$^+$15b]. Based on an earlier version of the two aforementioned works, Sangiovanni-Vincentelli *et al.* describe a methodology that aims at merging contract-based design with the platform-based design paradigm in order to yield a meet-in-the-middle development approach [SVDP12]. In addition to the above generic contract frameworks that can be instantiated for different properties, there are also specialized approaches, e.g., the work of Tørngren *et al.* [TTDL12] or the OCRA framework [CDT13] that both focus on temporal properties.

There exist a number of approaches for collaborative editing of models, but which do not provide support to define iterative development processes supported by background verification. On the one hand, this includes data-base backends for the Eclipse Modeling Framework [Eclb], such as CDO [Ecla], Teneo [Ecld] or NeoEMF [Neo]. On the other hand, the model repository EMFStore [Eclc] enables collaborative editing and supports change management based on model merging.

12.1.1 Problems Not Solved in DREAMS

D.1 **Adaptivity and System Collaboration (with open-world assumption).** The development process that has been employed in DREAMS is based on a sequential transition of the V-model, which a focus on the design and implementation steps defined in the left wing. The process is intended to deploy a set of applications to an instance of the DREAMS platform, both known and fixed at design-time (closed-world assumption). Based on a selected fault-model, the DREAMS development process can be used to derive failure mitigation strategies (e.g., system reconfiguration based on component-to-platform mappings and resource schedules pre-computed at design-time).

In DREAMS, the considered set of applications under development is regarded in isolation, i.e., uncertainty and emerging behavior based on environmental inputs (e.g., sensor data) or run-time binding with other DREAMS systems is not foreseen. Likewise, also the structure and configuration of the DREAMS platform relies solely on considerations taken at design-time.

Although MDE plays an essential role to mitigate the complexity of the DREAMS platform, e.g., to derive correct and consistent platform configuration artifacts, further concepts are needed to consider run-time changes of the system or its environment (adaptivity), and to include the possibility to collaborate with other systems into the development process. Here, the main challenge is to ensure the absence of undesired behavior at both the level of the constituent systems, as well as the SoS, and to provide concepts to maintain certifiability.

D.2 **Iterative exploration process supported by continuous verification.** The DREAMS MDE process follows the V-model in order to satisfy the requirements posed by safety standards. Based on a requirements specification, and a functional description, the architecture of the system is defined, i.e. the logical application sub-systems and their inter-connections, and a coarse-grained model of the target platform (including the hardware and the middleware). In the module design step, the sub-systems at the application level are refined into atomic components, whereas the platform model is refined until the most-fine grained level considered in the MDE process is reached (e.g., hypervisor partitions, processor cores). Only after the refinement has been finalized, verification is performed in a bottom-up manner, i.e. at the module level, the system design level, for the software architecture, and finally for the system requirements. As a consequence, to verify for instance timing requirements, the design must be completed, such that the refinement chain including deployment, resource allocation and possibly also configuration generation can be traversed. Only then, the requirements specified by the system designer (e.g., end-to-end constraints) and derived constraints (e.g., load of specific resources) can be checked.

Although in the DREAMS MDE process the individual refinement and verification steps are automated, and also although models are used to estimate the quality of designs in early phases of the process (e.g., for design-space exploration), each of the steps has to be triggered manually by the system engineer at well-defined synchronization points in the development process. In contrast to that, the idea of the continuous model-based verification approach sketched in Section 12.1 is to apply all possible refinement and verification steps as early as possible (i.e., also on underspecified system models) in an automated manner, in order to provide feedback as soon as possible.

12.2 Innovation for Road-map

I.1 **Understand requirements for MDE process for mixed-critical adaptive systems.** In order to be able to define an MDE process for adaptive systems, first the additional requirements onto the development that stem from the open-world assumption and the adaptive behavior of the MCS under development have to be analyzed.

On the one hand, this concerns additional requirements onto the underlying metamodels that need to be extended with means to describe the behavior of the environment (that would trigger the adaptive behavior), as well as the assumptions that need to be met to ensure the correct operation of the system under development. On the other hand, this also concerns the question how based on the aforementioned models of assumptions onto systems inputs, and models of the guarantees provided by the system, parts of the system development process can be shifted to run-time, while meeting the certification requirements for MCS.

I.2 **Identify and define models for collaborative behavior.** Considering the interactions between the constituent adaptive MCS in an SoS requires appropriate models to describe the involved interfaces, including the assumptions made by the systems' services, and the guarantees that they can be provide.

For instance, in the autonomous car scenario, a platoon can only be established in case the car running ahead provides a distance control signal with sufficient quality of service guarantees (e.g., real-time properties, or the assurance that the signal is gathered by a (sub-)system that is certified to the corresponding safety integrity level). The example illustrates that on the one hand, the description of service assumptions and guarantees must cover properties in different dimensions that are relevant for MCS (e.g., real-time behavior, safety, and security). On the other hand, models for collaborative behavior have to be applicable at both design-time (to allow predicting the behavior of an SoS based on the properties of its constituent systems, i.e. compositionality [GS05]) and at run-time (as a basis for online verification of the assumptions of the SoS).

I.3 **Understand underspecified artifacts, their inherent constraints and feasible assumptions.** A prerequisite to define a process for iterative model-driven engineering of MCS is to understand for which aspects of the system underspecified artifacts could be used. Possible candidates are artifacts from all viewpoints introduced in the DREAMS MDE approach [BDM+17a], including logical components (where black-box components or incomplete interface specifications might evolve during the development process), nodes of the platform model (that might be dimensioned by adding resources or parametrized when refining the system specification), or even platform-specific artifacts such as time slots in a time-triggered schedule (for which initially only bounds on the start and end time could be determined that are subsequently refined).

In a first step, it must be analyzed how underspecified artifacts can be described, and what is their semantics. A second step is to derive which inherent constraints are imposed on underspecified artifacts, in particular, when used in a certain context of the system model. Further it must be investigated what are suitable mechanisms to describe the dependencies between (possibly underspecified) artifacts, and how these can be used to support automatic model refinement. Lastly, another important question is how the intend of the system engineer can be intuitively taken into account in such highly automated process.

I.4 **Develop synthesis methods to evolve underspecified models in accordance with constraints and assumptions.** Collaborative, iterative MDE relies on tool support that provides immediate feedback to system engineers in case the design is changed. In this context, methods for the automatic refinement of underspecified models need to be developed that can be used to generate different design alternatives considering the results from automated model quality estimation and constraint verification methods.

For instance, in case the logical model of an application is refined (e.g., by adding a new component, or providing a more detailed model of a coarse-grained component), feasible deployments to the platform model should be determined automatically. After the system designer has selected an appropriate component-to-platform mapping, further dependent artifacts such as resource schedules should be derived.

Lastly, the tools implementing the synthesis methods used to refine underspecified system descriptions should also be compatible with techniques that enable to collaboratively work on the underlying model (c.f. discussion of existing frameworks at the end of Section 12.1).

Bibliography

[aar] aaron. Cubietech cubieboard2.

[ABD$^+$13] Andreas Abel, Florian Benz, Johannes Doerfert, Barbara Dörr, Sebastian Hahn, Florian Haupenthal, Michael Jacobs, Amir H. Moin, Jan Reineke, Bernhard Schommer, and Reinhard Wilhelm. Impact of resource sharing on performance and performance prediction: A survey. In *Proc. 24th International Conference, CONCUR 2013, Buenos Aires, Argentina, August 27-30.*, pages 25–43, 2013.

[ACQ$^+$11] J. Abella, F. J. Cazorla, E. Quinones, A. Grasset, S. Yehia, P. Bonnot, D. Gizopoulos, R. Mariani, and G. Bernat. Towards improved survivability in safety-critical systems. In *IEEE 17th International On-Line Testing Symposium (IOLTS)*, pages 240–245, 2011.

[ACR14] Dean Michael Ancajas, Koushik Chakraborty, and Sanghamitra Roy. Fort-NoCs: Mitigating the Threat of a Compromised NoC. In *Proceedings of the 51st Annual Design Automation Conference*, DAC '14, pages 158:1–158:6. ACM, 2014.

[AFF$^+$17] Ankit Agrawal, Gerhard Fohler, Johannes Freitag, Jan Nowotsch, Sascha Uhrig, and Michael Paulitsch. Contention-Aware Dynamic Memory Bandwidth Isolation with Predictability in COTS Multicores: An Avionics Case Study. In Marko Bertogna, editor, *29th Euromicro Conference on Real-Time Systems (ECRTS 2017)*, volume 76 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 2:1–2:22, Dagstuhl, Germany, 2017. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

[AFS97] William A Arbaugh, David J Farber, and Jonathan M Smith. A secure and reliable bootstrap architecture. In *Security and Privacy, 1997. Proceedings., 1997 IEEE Symposium on*, pages 65–71. IEEE, 1997.

[AGRS06] T. Abdelzaher, C.D. Gill, R. Rajkumar, and J.A. Stankovic. Distributed real-time and embedded systems research in the context of GENI. Technical report, National Science Foundation Workshop, 2006.

[ALMW09] Jesper Andersson, Rogério Lemos, Sam Malek, and Danny Weyns. Modeling dimensions of self-adaptive software systems. In Betty H. C. Cheng, Rogério Lemos, Holger Giese, Paola Inverardi, and Jeff Magee, editors, *Software Engineering for Self-Adaptive Systems*, pages 27–47, Berlin, Heidelberg, 2009. Springer.

[ALRL04] Algirdas Avizienis, J-C Laprie, Brian Randell, and Carl Landwehr. Basic concepts and taxonomy of dependable and secure computing. *Dependable and Secure Computing, IEEE Transactions on*, 1(1):11–33, 2004.

[APB$^+$17] Mohammad Ashjaei, Gaetano Patti, Moris Behnam, Thomas Nolte, Giuliana Alderisi, and Lucia Lo Bello. Schedulability analysis of ethernet audio video bridging networks with scheduled traffic support. *Real-Time Systems*, 53(4):526–577, July 2017.

[ARI05] ARINC. ARINC Specification 653-2, Avionics Application Software Standard Interface, December 2005.

[ATED14] Philip Axer, Daniel Thiele, Rolf Ernst, and Jonas Diemer. Exploiting shaper context to improve performance bounds of ethernet avb networks. In *2014 51st ACM/EDAC/IEEE Design Automation Conference (DAC)*, 2014.

[Ava15] Arianna Avanzini. Integrating Linux and the real-time ERIKA OS through the Xen hypervisor. *Industrial Embedded Systems (SIES)*, 2015.

[AWHJ15] O. Arias, J. Wurm, K. Hoang, and Y. Jin. Privacy and security in internet of things and wearable devices. *IEEE Transactions on Multi-Scale Computing Systems*, 1(2):99–109, April 2015.

[BAEP14] Unmesh D. Bordoloi, Amir Aminifar, Petru Eles, and Zebo Peng. Schedulability analysis of ethernet AVB switches. In *2014 IEEE 20th International Conference on Embedded and Real-Time Computing Systems and Applications*, 2014.

[BBBT11] C. Baumann, T. Bormer, H. Blasum, and S. Tverdyshev. Proving memory separation in a microkernel by code level verification. In *14th IEEE International Symposium on Object/Component/Service-Oriented Real-Time Distributed Computing Workshops*, pages 25–32, 3 2011.

[BCN+15a] Albert Benveniste, Benoît Caillaud, Dejan Nickovic, Roberto Passerone, Jean-Baptiste Raclet, Philipp Reinkemeier, Alberto Sangiovanni-Vincentelli, Werner Damm, Tom Henzinger, and Kim Larsen. Contracts for systems design: Methodology and application cases. Research Report RR-8760, Inria Rennes Bretagne Atlantique, July 2015.

[BCN+15b] Albert Benveniste, Benoît Caillaud, Dejan Nickovic, Roberto Passerone, Jean-Baptiste Raclet, Philipp Reinkemeier, Alberto Sangiovanni-Vincentelli, Werner Damm, Tom Henzinger, and Kim Larsen. Contracts for systems design: Theory. Research Report RR-8759, Inria Rennes Bretagne Atlantique, July 2015.

[BCNP12] Frédéric Boniol, Hugues Cassé, Eric Noulard, and Claire Pagetti. Deterministic execution model on COTS hardware. In Andreas Herkersdorf, Kay Römer, and Uwe Brinkschulte, editors, *Architecture of Computing Systems – ARCS 2012: 25th International Conference, Munich, Germany, February 28 - March 2, 2012. Proceedings*, pages 98–110. Springer, 2012.

[BD16] Alan Burns and Robert I. Davis. Mixed criticality systems - a review. Report, University of York, 2016.

[BDM93] M. Barborak, A. Dahbura, and Miroslaw Malek. The consensus problem in fault-tolerant computing. *ACM Comput. Surv.*, 25:171–220, June 1993.

[BDM+17a] Simon Barner, Alexander Diewald, Jörn Migge, Ali Syed, Gerhard Fohler, Madeleine Faugère, and Daniel Gracia Pérez. DREAMS toolchain: Model-driven engineering of mixed-criticality systems. In *Proceedings of the ACM/IEEE 20th International Conference on Model Driven Engineering Languages and Systems (MODELS '17)*, Austin, TX, USA, September 2017. To appear.

[BDM+17b] Simon Barner, Alexander Diewald, Jörn Migge, Ali Abbas Jaffari Syed, Gerhard Fohler, Faugére Madeleine, and Daniel Gracia Pérez. DREAMS Toolchain: Model-Driven Engineering of Mixed-Criticality Systems. In *ACM/IEEE 20th International Conference on Model Driven Engineering Languages and Systems*, September 2017.

[BFB15] A. Burns, T. Fleming, and S. Baruah. Cyclic executives, multi-core platforms and mixed criticality applications. In *2015 27th Euromicro Conference on Real-Time Systems*, pages 3–12, July 2015.

[BFGP03] A. Bobbio, G. Franceschinis, Rossano Gaeta, and Luigi Portinale. Parametric fault tree for the dependability analysis of redundant systems and its high-level petri net semantics. *IEEE Trans. Softw. Eng.*, 29:270–287, March 2003.

[BGF+08] Nelly Bencomo, Paul Grace, Carlos Flores, Danny Hughes, and Gordon Blair. Genie: Supporting the model driven development of reflective, component-based adaptive systems. In *Proceedings of the 30th International Conference on Software Engineering*, ICSE '08, pages 811–814, New York, NY, USA, 2008. ACM.

[BGP+15] S. Berger, K. Goldman, D. Pendarakis, D. Safford, E. Valdez, and M. Zohar. Scalable attestation: A step toward secure and trusted clouds. In *2015 IEEE International Conference on Cloud Engineering*, pages 185–194, March 2015.

[BH08] Mark A. Bedau and Paul Humphreys. *Emergence: Contemporary Readings in Philosophy and Science*. MIT Press, 2008.

[BHRV13] Twan Basten, Roelof Hamberg, Frans Reckers, and Jacques Verriet, editors. *Model-Based Design of Adaptive Embedded Systems*, volume 22 of *Embedded Systems*. Springer, 2013.

[BO16] Jacques Brygier and Mehmet Oezer. Safety and security for the internet of things. Report, 2016.

[Bon16] Andrea Bondavalli, editor. *Cyber-physical systems of systems: Foundations - a conceptual model and some derivations: the AMADEOS legacy*, volume 10099 of *Lecture Notes in Computer Science*. Springer, Cham, 2016.

[BR10] Steve Balacco and Chris Rommel. Next generation embedded hardware architectures: Driving onset of project delays, costs overruns and software development challenges. Technical report, Klockwork, Inc., September 2010.

[BS] H. P. Breivold and K. Sandstrm. Internet of things for industrial automation – challenges and technical solutions. In *2015 IEEE International Conference on Data Science and Data Intensive Systems*, pages 532–539.

[CAA+16] Francisco J Cazorla, Jaume Abella, Jan Andersson, Tullio Vardanega, Francis Vatrinet, Iain Bate, Ian Broster, Mikel Azkarate-Askasua, Franck Wartel, Liliana Cucu, et al. PROXIMA: Improving measurement-based timing analysis through randomisation and probabilistic analysis. In *Digital System Design (DSD), 2016 Euromicro Conference on*, pages 276–285. IEEE, 2016.

[CCP+15] B. Cilku, A. Crespo, P. Puschner, J. Coronel, and S. Peiro. A tdma-based arbitration scheme for mixed-criticality multicore platforms. In *International Conference on Event-based Control, Communication, and Signal Processing (EBCCSP), pp: 17-19, 2015. Krakow. Poland*, 2015.

[CCR+17] F. Ciccozzi, I. Crnkovic, D. Di Ruscio, I. Malavolta, P. Pelliccione, and R. Spalazzese. Model-driven engineering for mission-critical iot systems. *IEEE Software*, 34(1):46–53, January 2017.

[CdLG+09] B. Cheng, R. de Lemos, H. Giese, P. Inverardi, J. Magee, J. Andersson, B. Becker, N. Bencomo, Y. Brun, B. Cukic, et al. Software engineering for self-adaptive systems: A research roadmap. In *Software Engineering for Self-Adaptive Systems*, Lecture Notes in Computer Science, pages 1–26. Springer, Dagstuhl Castle, Germany, 2009.

[CDT13] Alessandro Cimatti, Michele Dorigatti, and Stefano Tonetta. OCRA: A tool for checking the refinement of temporal contracts. In *Proceedings of the 28th IEEE/ACM International Conference on Automated Software Engineering*, ASE'13, pages 702–705, Piscataway, NJ, USA, 2013. IEEE Press.

[Cer10] M. Cernak. A comparison of decision tree classifiers for automatic diagnosis of speech recognition errors. *Computing and Informatics*, 29(3):489–501, 2010.

[CER13] FP7 CERTAINTY. Certification of real-time applications designed for mixed-criticality (CERTAINTY), 2013.

[Cha06] David J. Chalmers. *The Re-Emergence of Emergence, Strong and Weak Emergence*. Oxford University Press, 2006.

[CKR+12] S. Chattopadhyay, C. L. Kee, A. Roychoudhury, T. Kelter, P. Marwedel, and H. Falk. A unified WCET analysis framework for multi-core platforms. In *2012 IEEE 18th Real Time and Embedded Technology and Applications Symposium*, pages 99–108, April 2012.

[CMK+11] Stephen Checkoway, Damon McCoy, Brian Kantor, Danny Anderson, Hovav Shacham, Stefan Savage, Karl Koscher, Alexei Czeskis, Franziska Roesner, Tadayoshi Kohno, et al. Comprehensive experimental analyses of automotive attack surfaces. In *USENIX Security Symposium*. San Francisco, 2011.

[Con11] M. Concepcion. *Diagnostics Strategies of Modern Automotive Systems*. CreateSpace, 2011.

[CON14] FP7 CONTREX. Design of embedded mixed-criticality CONTRol systems under consideration of EXtra-functional properties (CONTREX), 2014.

[CQnV+13] Francisco J. Cazorla, Eduardo Quiñones, Tullio Vardanega, Liliana Cucu, Benoit Triquet, Guillem Bernat, Emery Berger, Jaume Abella, Franck Wartel, Michael Houston, Luca Santinelli, Leonidas Kosmidis, Code Lo, and Dorin Maxim. PROARTIS: Probabilistically Analyzable Real-Time Systems. *ACM Trans. Embed. Comput. Syst.*, 12(2s):94:1–94:26, May 2013.

[CS17] Federico Ciccozzi and Romina Spalazzese. MDE4IoT: Supporting the internet of things with model-driven engineering. In *Intelligent Distributed Computing X: Proceedings of the*

10th International Symposium on Intelligent Distributed Computing – IDC 2016, Paris, France, October 10-12 2016, pages 67–76, Cham, 2017. Springer.

[CSB+17] A. Crespo, A. Soriano, P. Balbastre, J. Coronel, D. Gracia, and P. Bonnot. Hypervisor feedback control of mixed critical systems: the xtratum approach. In *Proceedings of the Workshop on Operating Systems Platforms for Embedded Real-Time Applications (OSPERT). Dubrovnik June 2017.*, 2017.

[DFG+14] Guy Durrieu, Madeleine Faugère, Sylvain Girbal, Daniel Gracia Pérez, Claire Pagetti, and Wolfgang Puffitsch. Predictable flight management system implementation on a multicore processor. In *Embedded Real Time Software and Systems (ERTS^2)*, volume 14, page 1, Toulouse, France, 2014.

[DFG+16] Guy Durrieu, Gerhard Fohler, Gautam Gala, Sylvain Girbal, Daniel Gracia Pérez, Eric Noulard, Claire Pagetti, and Simara Pérez. DREAMS about reconfiguration and adaptation in avionics. In *Embedded Real Time Software and Systems (ERTS^2)*, Toulouse, France, January 2016.

[DFH+03] B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, I. Pratt, A. Warfield, P. Barham, and R. Neugebauer. Xen and the art of virtualization. In *Proceedings of the ACM Symposium on Operating Systems Principles*, October 2003.

[DHR93] D. Driankov, H. Hellendoorn, and M. Reinfrank. *An introduction to fuzzy control*. Springer-Verlag New York, Inc., New York, NY, USA, 1993.

[dLGM+13] Rogério de Lemos, Holger Giese, Hausi A. Müller, Mary Shaw, Jesper Andersson, Marin Litoiu, Bradley Schmerl, Gabriel Tamura, Norha M. Villegas, Thomas Vogel, Danny Weyns, Luciano Baresi, Basil Becker, Nelly Bencomo, Yuriy Brun, Bojan Cukic, Ron Desmarais, Schahram Dustdar, Gregor Engels, Kurt Geihs, Karl M. Göschka, Alessandra Gorla, Vincenzo Grassi, Paola Inverardi, Gabor Karsai, Jeff Kramer, Antónia Lopes, Jeff Magee, Sam Malek, Serge Mankovskii, Raffaela Mirandola, John Mylopoulos, Oscar Nierstrasz, Mauro Pezzè, Christian Prehofer, Wilhelm Schäfer, Rick Schlichting, Dennis B. Smith, João Pedro Sousa, Ladan Tahvildari, Kenny Wong, and Jochen Wuttke. Software engineering for self-adaptive systems: A second research roadmap. In *Software Engineering for Self-Adaptive Systems II: International Seminar, Dagstuhl Castle, Germany, October 24-29, 2010 Revised Selected and Invited Papers*, pages 1–32. Springer, 2013.

[DRE13] H2020 DREAMS. Distributed real-time architecture for mixed criticality systems (DREAMS), 2013.

[DRE14] DREAMS Consortium. Architectural Style of DREAMS, 2014. Deliverable D 1.2.1.

[DRIP11] Davide Di Ruscio, Ludovico Iovino, and Alfonso Pierantonio. What is needed for managing co-evolution in MDE? In *Proceedings of the 2nd International Workshop on Model Comparison in Practice*, IWMCP '11, pages 30–38, New York, NY, USA, 2011. ACM.

[DTE12] Jonas Diemer, Daniel Thiele, and Rolf Ernst. Formal worst-case timing analysis of ethernet topologies with strict-priority and AVB switching. In *7th IEEE International Symposium on Industrial Embedded Systems (SIES'12)*, 2012.

[EAS11] EASA. Development assurance of airborne electronic hardware, 2011.

[eas12] MULCORS - use of multicore processors in airborne systems (research project easa.2011/6). Technical report, EASA, 16th December 2012.

[EAS13] EASA. Certification memorandum - software aspects of certification - easa. Technical report, 9th March 2013.

[Ecla] Eclipse. CDO model repository. https://www.eclipse.org/cdo/.

[Eclb] Eclipse. Eclipse modeling framework (EMF). http://www.eclipse.org/emf/.

[Eclc] Eclipse. EMFStore model repository. http://www.eclipse.org/emfstore/.

[Ecld] Eclipse. Teneo: Model-relational database integration. https://wiki.eclipse.org/Teneo.

[(Ed11] R. Obermaisser (Ed.). *Time-Triggered Communication*. CRC Press, Taylour & Francis Group. 568 pages, ISBN 9781439846612., 2011.

[EJS+10] Christian Engel, Eric Jenn, Peter H Schmitt, Rodrigo Coutinho, and Tobias Schoofs. Enhanced dispatchability of aircrafts using multi-static configurations. In *Embedded Real Time Software and Systems (ERTSS'10)*, 2010.

[Elh11] M. Elhadef. Solving the pmc-based system-level fault diagnosis problem using hopfield neural networks. In *Proc. of IEEE Int. Conf. on Adv. Information Networking and Applications*, pages 216–223, 2011.

[EMC14] ARTEMIS EMC2. EMC2 - Embedded Multi-Core systems for Mixed-Criticality applications in dynamic and changeable real-time environments, 2014.

[ENNT15] Alexandre Esper, Geoffrey Nelissen, Vincent Nélis, and Eduardo Tovar. How realistic is the mixed-criticality real-time system model? In *Proceedings of the 23rd International Conference on Real Time and Networks Systems*, pages 139–148. ACM, 2015.

[Ern10] Rolf Ernst. Certification of trusted MPSoC platforms. In *MPSoC Forum*, 2010.

[EUR92] EUROCAE. DO-178B: Software Considerations in Airborne Systems and Equipment Certification, 1992.

[EUR00] EUROCAE. DO-254: Design assurance guidance for airborne electronic hardware, 2000.

[EUR01] EUROCAE. DO-297: Software, Electronic, Integrated Modular Avionics (IMA) Development Guidance and Certification Considerations, 2001.

[Eva15] Evan Perez. FBI: Hacker claimed to have taken over flight's engine controls. http://edition.cnn.com/2015/05/17/us/fbi-hacker-flight-computer-systems/index.html, 2015.

[Fis14] Stuart Fisher. Whitepaper: Certifying Applications in a Multi-Core Environment. Technical report, SYSGO, January 2014.

[FOu] Linux FOundation. The xen project, the powerful open source industry standard for virtualization.

[FRC$^+$13] N. Ferry, A. Rossini, F. Chauvel, B. Morin, and A. Solberg. Towards model-driven provisioning, deployment, monitoring, and adaptation of multi-cloud systems. In *2013 IEEE Sixth International Conference on Cloud Computing*, pages 887–894, June 2013.

[Fri94] B. Fritzke. Fast learning with incremental RBF networks. In *Neural Processing Letters*, pages 2–5, 1994.

[Fuc10] R. Fuchsen. How to address certification for multi-core based IMA platforms: Current status and potential solutions. In *IEEE/AIAA 29th Digital Avionics Systems Conference (DASC)*, 2010.

[GAdSP14] Stefan Groesbrink, Luis Almeida, Mario de Sousa, and Stefan M. Petters. Towards certifiable adaptive reservations for hypervisor-based virtualization. In *20th IEEE Real-Time and Embedded Technology and Applications Symposium, RTAS 2014, Berlin, Germany, April 15-17, 2014*, pages 13–24, 2014.

[Gef] Jason Geffner. Venom virtualized environment neglected operations manipulation.

[GH10] Kees Goossens and Andreas Hansson. The aethereal network on chip after ten years: goals, evolution, lessons, and future. In *Proceedings of the 47th Design Automation Conference, DAC 2010, Anaheim, California, USA, July 13-18, 2010*, pages 306–311, 2010.

[GJLR$^+$15] Sylvain Girbal, Xavier Jean, Jimmy Le Rhun, Daniel Gracia Pérez, and Marc Gatti. Deterministic Platform Software for Hard Real-Time Systems Using Multi-Core COTS. In *IEEE/AIAA 32nd Digital Avionics Systems Conference (DASC), 05 - 10 Oct 2013, East Syracuse, NY, USA*. IEEE, 2015.

[GJR$^+$15] Sylvain Girbal, Xavier Jean, Jimmy Le Rhun, Daniel Gracia Pérez, and Marc Gatti. Deterministic platform software for hard real-time systems using multi-core cots. In *34th Digital Avionics Systems Conference (DASC'2015), Prague, Czech Republic, 2015*, 2015. Thales Research & Technology.

[GKGP$^+$16] Gautam Gala, Thomas Koller, Daniel Gracia Pérez, Gerhard Fohler, and Christoph Ruland. Timing analysis of secure communication between resource managers in DREAMS. In *Proceedings of the Workshop on Security and Dependability of Critical Embedded Real-Time Systems*. 1st Workshop on Security and Dependability of Critical Embedded Real-Time Systems in conjunction with RTSS 16, November 2016.

[Gri06] A.-M. Grisogono. The implications of complex adaptive systems theory for C2. Technical report, Defence Science and Technology Organisation, 2006.

[GS05] Gregor Gssler and Joseph Sifakis. Composition for component-based modeling. *Science of Computer Programming*, 55(1):161–183, 2005. Formal Methods for Components and Objects: Pragmatic aspects and applications.

[GSHT13] Georgia Giannopoulou, Nikolay Stoimenov, Pengcheng Huang, and Lothar Thiele. Scheduling of mixed-criticality applications on resource-sharing multicore systems. In *Proc. of the Int. Conf. on Embedded Software, EMSOFT 2013, Montreal, QC, Canada*, pages 17:1–17:15, 2013.

[Ham03] Rob Hammett. Flight-critical distributed systems - Design considerations [avionics]. *IEEE Aerospace and Electronic Systems Magazine*, 18(6):30–36, 2003.

[HD03] F. Hutter and R. Dearden. Efficient on-line fault diagnosis for non-linear systems, 2003.

[HE16] Gernot Heiser and Kevin Elphinstone. L4 microkernels: The lessons from 20 years of research and deployment. *ACM Trans. Comput. Syst.*, 34(1):1:1–1:29, 2016.

[HHL+97] Hermann Härtig, Michael Hohmuth, Jochen Liedtke, Sebastian Schönberg, and Jean Wolter. The performance of μkernel-based systems. In *SOSP*, pages 66–77, 1997.

[HJ14] Sanghyun Han and Hyun-Wook Jin. Resource partitioning for integrated modular avionics: comparative study of implementation alternatives. *Softw., Pract. Exper.*, 44(12):1441–1466, 2014.

[HKB+09] Jens Happe, Heiko Koziolek, Umesh Bellur, Holger Giese, Wilhelm Hasselbring, Robert Laddaga, Margaria Tiziana, Josu Martinez, Christian Müller-Schloer, and Roland Reichle. The role of models in self-adaptive and self-healing systems. In Artur Andrzejak, Kurt Geihs, Onn Shehory, and John Wilkes, editors, *Self-Healing and Self-Adaptive Systems*, number 09201 in Dagstuhl Seminar Proceedings. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, Germany, 2009.

[HM01] Ellis F. Hitt and Dennis Mulcar. *Fault-Tolerant Avionics*, chapter 28. CRC Press LLC, 2001.

[HSF16] Florian Heilmann, Ali Abbas Jaffari Syed, and Gerhard Fohler. Mode-changes in COTS time-triggered network hardware without online reconfiguration. In *14th Workshop on Real-Time Networks (RTN'16) in conjuction with 28th Euromicro International Conference on Real-time Systems (ECRTS'16)*, July 2016.

[IEC10a] IEC. IEC 61508-1: Functional safety of electrical/electronic/programmable electronic safety-related systems - part 1: General requirements, 2010.

[IEC10b] IEC. IEC 61508-2: Functional safety of electrical/electronic/programmable electronic safety-related systems - part 2: Requirements for electrical / electronic / programmable electronic safety-related systems, 2010.

[IEC10c] IEC. IEC 61508-3: Functional safety of electrical/electronic/programmable electronic safety-related systems - part 3: Software requirements, 2010.

[Inf] Information Society and Media Directorate-General Unit G3/Computing Systems Research Objective . Report from the Workshop on Mixed Criticality Systems held on 03rd February 2012 in Brussels, Belgium . http://cordis.europa.eu/fp7/ict/embedded-systems-engineering/documents/sra-mixed-criticality-systems.pdf.

[INT] INTER-IoT H2020 Project. Interoperability of Heterogeneous IoT Platforms. http://www.inter-iot-project.eu/.

[Ise06] R. Isermann. *Fault-diagnosis systems: an introduction from fault detection to fault tolerance*. Springer, 2006.

[IT09] Paola Inverardi and Massimo Tivoli. *Software Engineering: International Summer Schools (ISSSE)*, chapter The Future of Software: Adaptation and Dependability, pages 1–31. Springer, Berlin, Heidelberg, 2009.

[Iti07] Jean-Bernard Itier. A380 integrated modular avionics. In *Proceedings of the ARTIST2 meeting on integrated modular avionics*, volume 1, pages 72–75, 2007.

[JKT08] T. Jiang, K. Khorasani, and S. Tafazoli. Parameter estimation-based fault detection, isolation and recovery for nonlinear satellite models. *IEEE Transactions on Control Systems Technology*, 16(4):799–808, July 2008.

[KCR+10] Karl Koscher, Alexei Czeskis, Franziska Roesner, Shwetak Patel, Tadayoshi Kohno, Stephen Checkoway, Damon McCoy, Brian Kantor, Danny Anderson, Hovav Shacham, et al. Experimental security analysis of a modern automobile. In *Security and Privacy (SP), 2010 IEEE Symposium on*, pages 447–462. IEEE, 2010.

[KGP+16] T. Koller, G. Gala, D. Gracia Pérez, C. Ruland, and G. Fohler. DREAMS: Secure communication between resource management components in networked multi-core systems. In *2016 IEEE Conference on Open Systems (ICOS)*, pages 99–104, October 2016.

[KHF+15] H. Kopetz, O. Höftberger, B. Frömel, F. Brancati, and A. Bondavalli. Towards an understanding of emergence in systems-of-systems. In *2015 10th System of Systems Engineering Conference (SoSE)*, pages 214–219, May 2015.

[KHM+16] Tim Kelly, Ibrahim Habli, Paulo Masiero, Andr Oliveira, Rosana T.V. Braga, and Yiannis Papadopoulos. Model-based safety analysis of software product lines. 8:412, 01 2016.

[KHMF13] Timon Kelter, Tim Harde, Peter Marwedel, and Heiko Falk. Evaluation of resource arbitration methods for multi-core real-time systems. In *OASIcs – OpenAccess Series in Informatics*, volume 30. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2013.

[KJE+17] Carlos Kamienski, Marc Jentsch, Markus Eisenhauer, Jussi Kiljander, Enrico Ferrera, Peter Rosengren, Jesper Thestrup, Eduardo Souto, Walter S. Andrade, and Djamel Sadok. Application development for the internet of things: A context-aware mixed criticality systems development platform. *Computer Communications*, 104:1–16, 2017.

[KJK15] D. Kim, Y. Jeon, and J. Kim. A method based on platform integrity verification for activating a mobile trusted module. In *2015 International Conference on Information and Communication Technology Convergence (ICTC)*, pages 1174–1176, October 2015.

[KM03] N.Y. Kuznetsov and K.V. Mikhalevich. Reliability analysis of systems described by fault trees with efficiency. *Cybernetics and Sys. Anal.*, 39:746–752, September 2003.

[KNH+97] H. Kopetz, R. Nossal, R. Hexel, A. Krger, D. Millinger, R. Pallierer, C. Temple, and M. Krug. Mode handling in the time-triggered architecture. *IFAC Proceedings Volumes*, 30(15):11–16, 1997. IFAC Workshop on Distributed Computer Control Systems (DCCS'97), Seoul, Korea, 28-30 July 1997.

[KNP+13] Ondrej Kotaba, Jan Nowotsch, Michael Paulitsch, Stefan M. Petters, and Henrik Theilingx. Multicore in real-time systems - temporal isolation challenges due to shared resources. In *Workshop on Industry-Driven Approaches for Cost-effective Certification of Safety-Critical, Mixed-Criticality Systems (WICERT)*, 2013.

[KOESH07] H. Kopetz, R. Obermaisser, C. El Salloum, and B. Huber. Automotive software development for a multi-core system-on-a-chip. In *Fourth International Workshop on Software Engineering for Automotive Systems (ICSE Workshops SEAS)*, pages 2–9, 2007.

[KPR^{+}14] Angeliki Kritikakou, Claire Pagetti, Christine Rochange, Matthieu Roy, Madeleine Faugère, Sylvain Girbal, and Daniel Gracia Pérez. Distributed run-time WCET controller for concurrent critical tasks in mixed-critical systems. In *Proceedings of the 22th International Conference on Real-Time and Network Systems (RTNS'14)*, pages 139–148, 2014.

[KRF^{+}14] Angeliki Kritikakou, Christine Rochange, Madeleine Faugère, Claire Pagetti, Matthieu Roy, Sylvain Girbal, and Daniel Gracia Pérez. Distributed run-time WCET controller for concurrent critical tasks in mixed-critical systems. In *22nd International Conference on Real-Time Networks and Systems, RTNS 2014, Versaille, France, October 8-10, 2014*, page 139, 2014.

[KSM08] P.M. Khilar, J.K. Singh, and Sudipta Mahapatra. Design and evaluation of a failure detection algorithm for large scale ad hoc networks using cluster based approach. In *Proc. of Int. Conf. on Information Tech*, pages 153–158, 2008.

[KVF17] Kristin Krüger, Marcus Völp, and Gerhard Fohler. Improving security for time-triggered real-time systems against timing inference based attacks by schedule obfuscation. In *Work-in-Progress Proceedings ECRTS'17*, June 2017.

[LAO^{+}16] A. Larrucea, H. Ahmadian, R. Obermaisser, J. Perez, and C. F. Nicolas. A realistic approach to a network-on-chip cross-domain pattern. In *Euromicro Conference on Digital System Design (DSD)*, pages 396–403, Aug 2016.

[Lar17] Asier Larrucea. *Development and Certification of Dependable Mixed-Criticality Embedded Systems*. Phd, 2017.

[Lin] Linaro. Op-tee.

[LMB^{+}17] Asier Larrucea, Imanol Martinez, Vicent Brocal, Hamidreza Ahmadian, Salvador Peir, Roman Obermaisser, and Jon Perez. Reusable generic design patterns for mixed-criticality systems based on dreams. *Microprocessors and Microsystems*, 54:35–46, 2017.

[LMO^{+}] Asier Larrucea, Imanol Martinez, Roman Obermaisser, Jon Perez, and Carlos Fernando Nicolas. Modular development of dependable mixed-criticality embedded systems. In *20th Euromicro Conference on Digital System Design (DSD 2017)*, pages 417–426.

[LPO15a] Asier Larrucea, Jon Pérez, and Roman Obermaisser. A modular safety case for an IEC 61508 compliant generic COTS processor. In *15th IEEE International Conference on Computer and Information Technology, CIT 2015; 14th IEEE International Conference on Ubiquitous Computing and Communications, IUCC 2015; 13th IEEE International Conference on Dependable, Autonomic and Secure Computing, DASC 2015; 13th IEEE International Conference on Pervasive Intelligence and Computing, PICom 2015, Liverpool, United Kingdom, October 26-28, 2015*, pages 1788–1795, 2015.

[LPO15b] Asier Larrucea, Jon Perez, and Roman Obermaisser. A modular safety case for an iec61508 compliant generic cots multicore process. In *The 13th IEEE International Conference on Dependable, Autonomic and Secure Computing (DASC)*, 2015.

[LPO15c] Asier Larrucea, Jon Perez, and Roman Obermaisser. A modular safety case for an iec61508 compliant generic hypervisor. In *Euromicro Conference on Digital System Design (DSD)*, 2015.

[Ltd16] ARM Ltd. https://github.com/ARM-software/arm-trusted-firmware, 2016.

[LWVH12] Adam Lackorzyński, Alexander Warg, Marcus Völp, and Hermann Härtig. Flattening hierarchical scheduling. In *Proceedings of the Tenth ACM International Conference on Embedded Software*, EMSOFT '12, pages 93–102, New York, NY, USA, 2012. ACM.

[Mat11] R. Matinnejad. Agile model driven development: An intelligent compromise. In *2011 Ninth International Conference on Software Engineering Research, Management and Applications*, pages 197–202, August 2011.

[MBJ+09] Brice Morin, Olivier Barais, Jean-Marc Jezequel, Franck Fleurey, and Arnor Solberg. Models @ Run.time to support dynamic adaptation. *Computer*, 42(10):44–51, October 2009.

[mcs12] Mixed criticality systems. Report, European Comission, February 3 2012.

[MDB+12] Boris Motruk, Jonas Diemer, Rainer Buchty, Rolf Ernst, and Mladen Berekovic. Idamc: A many-core platform with run-time monitoring for mixed-criticality. In *High-Assurance Systems Engineering (HASE), 2012 IEEE 14th International Symposium on*, pages 24–31. IEEE, 2012.

[MEA+10] Malcolm S. Mollison, Jeremy P. Erickson, James H. Anderson, Sanjoy K. Baruah, and John A. Scoredos. Mixed-criticality real-time scheduling for multicore systems, 2010.

[Mil87] R. Milne. Strategies for diagnosis. *IEEE Trans. on Sys., Man & Cybernetics*, SMC-17(3):333–339, 1987.

[MNN+17] Cláudio Maia, Geoffrey Nelissen, Luis Nogueira, Luis Miguel Pinho, and Daniel Gracia Pérez. Schedulability Analysis for Global Fixed-Priority Scheduling of the 3-Phase Task Model. In *23rd IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA 2017)*. IEEE Computer Society, 2017.

[MNPGP16] Cláudio Maia, Luis Nogueira, Luis Miguel Pinho, and Daniel Gracia Pérez. A Closer Look into the AER Model. In *21st IEEE International Conference on Emerging Technologies & Factory Automation*. IEEE, 2016.

[MRCM09] Miguel Masmano, Ismael Ripoll, Alfons Crespo, and J Metge. Xtratum: a hypervisor for safety critical embedded systems. In *11th Real-Time Linux Workshop*, pages 263–272. Citeseer, 2009.

[MUL14] FP7 MULTIPartes. MULTIPartes – Multi-cores Partitioning for Trusted Embedded Systems, 2014.

[MV15] Charlie Miller and Chris Valasek. Remote exploitation of an unaltered passenger vehicle. http://illmatics.com/Remote%20Car%20Hacking.pdf, August 2015.

[MYPB14] S. Mohan, M. K. Yoon, R. Pellizzoni, and R. Bobba. Real-time systems security through scheduler constraints. In *2014 26th Euromicro Conference on Real-Time Systems*, pages 129–140, July 2014.

[Nat16] National Highway Traffic Safety Administration. NHTSA, 2016.

[NEL+17] Carlos Fernando Nicolas, Fernando Eizaguirre, Asier Larrucea, Simon Barner, Franck Chauvel, Goiuria Sagardui, and Jon Perez. Gsn support of mixed-criticality systems certification. In *Dependable Smart Embedded and Cyber-physical Systems and Systems-of-Systems (DECSoS 2017), 12th International ERCIM/EWICS/ARTEMIS Workshop on*, volume Workshop, pages 169–175. Springer, 2017.

[Neo] NeoEMF. Multi-backend EMF persistence framework. http://www.neoemf.com/.

[NLY+11] Jun Nakajima, Qian Lin, Sheng Yang, Min Zhu, Shang Gao, Mingyuan Xia, Peijie Yu, Yaozu Dong, Zhengwei Qi, Kai Chen, and Haibing Guan. Optimizing virtual machines using hybrid virtualization. In *Proceedings of the 2011 ACM Symposium on Applied Computing (SAC), TaiChung, Taiwan, March 21 - 24, 2011*, pages 573–578, 2011.

[NPB+14] J. Nowotsch, M. Paulitsch, D. Bühler, H. Theiling, S. Wegener, and M. Schmidt. Multi-core interference-sensitive WCET analysis leveraging runtime resource capacity enforcement. In *2014 26th Euromicro Conference on Real-Time Systems*, pages 109–118, July 2014.

[NST+13] Risto Nevalainen, Oscar Slotosch, Dragos Truscan, Uwe Kremer, and Vicky Wong. Impact of multicore platforms in hardware and software certification. In *Workshop on Industry-Driven Approaches for Cost-effective Certification of Safety-Critical, Mixed-Criticality Systems (WICERT)*, 2013.

[OMG12] OMG. Object constraint language (OCL). ISO/IEC 19507, January 2012.

[OPE14] FP7 OPENCOSS. OPENCOSS – Open Platform for EvolutioNary Certification of Safety-critical Systems, 2014.

[ORM] N. Odintsova, I. Rish, and S. Ma. Multi-fault diagnosis in dynamic systems.

[PAS] PASS BMWi Project. Platform for Automotive Apps Guaranteeing Security and Safety. http://www.pass-projekt.de/.

[PB14] Victoria Y. Pillitteri and Tanya L. Brewer. Guidelines for smart grid cybersecurity. Technical report, National Institute of Standards and Technology (NIST), September 2014.

[PBB+12] Claire Pagetti, Pierre Bieber, Julien Brunel, Kushal Gupta, Eric Noulard, Thierry Planche, François Vialard, Clément Ketchedji, Bernard Bésinet, and Philippe Despres. Reconfigurable IMA platform: from safety assessment to test scenarios on the SCARLETT demonstrator. In *Embedded Real Time Software (ERTS'12)*, 2012.

[PCR11] L. Portinale and D. Codetta-Raiteri. Using dynamic decision networks and extended fault trees for autonomous FDIr. In *Proc. of the IEEE Int. Conf. on Tools with Artificial Intelligence*, pages 480–484, November 2011.

[Per17] Jon Perez. Multicore devices for next generation safety systems. In *Xilinx Functional Safety Working Group (FSWG)*, 2017.

[PG74] Gerald J. Popek and Robert P. Goldberg. Formal requirements for virtualizable third generation architectures. *Commun. ACM*, 17(7):412–421, 1974.

[PG08] Claudia Pons and Diego Garcia. A lightweight approach for the semantic validation of model refinements. *Electronic Notes in Theoretical Computer Science*, 220(1):43–61, 2008. Proceedings of the Fourth Workshop on Model Based Testing (MBT 2008).

[PG16] Héctor Pérez and J. Javier Gutiérrez. Enabling data-centric distribution technology for partitioned embedded systems. *IEEE Trans. Parallel Distrib. Syst.*, 27(11):3186–3198, 2016.

[PGN$^+$14a] Jon Perez, David Gonzalez, Carlos Fernando Nicolas, Ton Trapman, and Jose Miguel Garate. A safety certification strategy for IEC 61508 compliant industrial mixed-criticality systems based on multicore partitioning. In *17th Euromicro Conference on Digital Systems Design (DSD)*, Verona, Italy, 2014.

[PGN$^+$14b] Jon Perez, David Gonzalez, Carlos Fernando Nicolas, Ton Trapman, and Jose Miguel Garate. A safety concept for a wind power mixed-criticality embedded system based on multicore partitioning. In *11th International Symposium - Functional Safety in Industrial Applications (TV Rheinland)*, Cologne, Germany, 2014.

[PLt00] R.J. Patton and C.J. Lopez-toribio. Soft computing approaches to fault diagnosis for dynamic systems: a survey. In *Proc. of the 4th IFAC Symposium SAFEPROCESS*, pages 298–311. Elsevier, 2000.

[PM99] R. Pigeau and C. McCann. Clarifying the concepts of control and of command. In *Proc. of the Command and Control Research and Technology Symposium*, 1999.

[PMC67] F.P. Preparata, G. Metze, and Robert T. Chien. On the connection assignment problem of diagnosable systems. *IEEE Transactions on Electronic Computers*, EC-16(6):848–854, 1967.

[PPG$^+$16] Sandro Pinto, Jorge Pereira, Tiago Gomes, Mongkol Ekpanyapong, and Adriano Tavares. Towards a trustzone-assisted hypervisor for real time embedded systems. *IEEE Computer Architecture Letters*, 2016.

[PPY$^+$15] R. Pellizzoni, N. Paryab, M. K. Yoon, S. Bak, S. Mohan, and R. B. Bobba. A generalized model for preventing information leakage in hard real-time systems. In *21st IEEE Real-Time and Embedded Technology and Applications Symposium*, pages 271–282, April 2015.

[PTH$^+$90] W.D. Potter, B.E. Tonn, M.R. Hilliard, G.E. Liepins, S. L. Purucker, and R. T. Goeltz. Diagnosis, parsimony, and genetic algorithms. In *Proc. of int. conf. on Industrial and engineering applications of artificial intelligence and expert systems*, pages 1–8. ACM, 1990.

[Rei87] R Reiter. A theory of diagnosis from first principles. *Artif. Intell.*, 32(1):57–95, April 1987.

[RGSZ14] Franz J. Rammig, Stefan Grösbrink, Katharina Stahl, and Yuhong Zhao. Designing self-adaptive embedded real-time software – towards system engineering of self-adaptation. In *Proceedings of the 2014 Brazilian Symposium on Computing Systems Engineering*, SBESC '14, pages 37–42, Washington, DC, USA, 2014. IEEE Computer Society.

[RMMG08] Sylvain Rougemaille, Frédéric Migeon, Christine Maurel, and Marie-Pierre Gleizes. Model driven engineering for designing adaptive multi-agents systems. In Alexander Artikis, Gregory M. P. O'Hare, Kostas Stathis, and George Vouros, editors, *Engineering Societies in the Agents World VIII: 8th International Workshop, ESAW 2007, Athens, Greece, October 22-24, 2007, Revised Selected Papers*, pages 318–332, Berlin, Heidelberg, 2008. Springer.

[RNH$^+$16] S. A. Rashid, G. Nelissen, D. Hardy, B. Akesson, I. Puaut, and E. Tovar. Cache-persistence-aware response-time analysis for fixed-priority preemptive systems. In *2016 28th Euromicro Conference on Real-Time Systems (ECRTS)*, pages 262–272, July 2016.

[Rus11] John M. Rushby. New challenges in certification for aircraft software. In *Proceedings of the 11th International Conference on Embedded Software, EMSOFT 2011, part of the Seventh Embedded Systems Week, ESWeek 2011, Taipei, Taiwan, October 9-14, 2011*, pages 211–218, 2011.

[RW14] Jan Reineke and Reinhard Wilhelm. Impact of resource sharing on performance and performance prediction. In *DATE 2014, Dresden, Germany, March 24-28,*, pages 1–2, 2014.

[SAB15] M. Sabt, M. Achemlal, and A. Bouabdallah. Trusted execution environment: What it is, and what it is not. In *2015 IEEE Trustcom/BigDataSE/ISPA*, volume 1, pages 57–64, August 2015.

[SAF17a] H2020 SAFEPOWER. Safe and secure mixed-criticality systems with low power requirements (SAFEPOWER), 2017.

[SAF17b] H2020 SAFURE. Safety and security by design for interconnected mixed-critical cyber-physical systems (SAFURE), 2017.

[SB07] J.W. Sheppard and S.G. Butcher. A formal analysis of fault diagnosis with d-matrices. *J. Electron. Test.*, 23:309–322, August 2007.

[SBR10] J. Schneider, Michael Bohn, and Robert Rbger. Migration of automotive real-time software to multicore systems: First steps towards an automated solution. In *22nd EUROMICRO Conference on Real-Time Systems*, 2010.

[SCO08] SCOPE Promoting Open Carrier Grade Base Platforms. *Virtualization: State of the Art*, April 2008. http://www.scope-alliance.org.

[SEH$^+$12] Christian El Salloum, Martin Elshuber, Oliver Hoftberger, Haris Isakovic, and Armin Wasicek. The ACROSS MPSoC – a new generation of multi-core processors designed for safety-critical embedded systems. In *Digital System Design (DSD), 2012 15th Euromicro Conference on*, pages 105–113, 2012.

[SG01] Sullivan S. and Slenski. G. Managing electrical connection systems and wire integrity on legacy aerospace vehicles, 2001.

[SK10] Udo Steinberg and Bernhard Kauer. Nova: a microhypervisor-based secure virtualization architecture. In *Proceedings of the 5th European conference on Computer systems*, pages 209–222. ACM, 2010.

[SM99] J. Swingler and J. W. McBride. The degradation of road tested automotive connectors. In *Forty-Fifth IEEE Holm Conference on Electrical Contacts*, pages 146–152, 1999.

[SPC$^+$11] A. Schranzhofer, R. Pellizzoni, J. J. Chen, L. Thiele, and M. Caccamo. Timing analysis for resource access interference on adaptive resource arbiters. In *2011 17th IEEE Real-Time and Embedded Technology and Applications Symposium*, pages 213–222, April 2011.

[SRWE08] Michele Sama, David S. Rosenblum, Zhimin Wang, and Sebastian Elbaum. Model-based fault detection in context-aware adaptive applications. In *Proceedings of the 16th ACM SIGSOFT International Symposium on Foundations of Software Engineering*, SIGSOFT '08/FSE-16, pages 261–271, New York, NY, USA, 2008. ACM.

[SS16] Johannes Specht and Soheil Samii. Urgency-based scheduler for time-sensitive switched ethernet networks. In *2016 28th Euromicro Conference on Real-Time Systems*, 2016.

[ST09] Mazeiar Salehie and LadaS Tahvildari. Self-adaptive software: Landscape and research challenges. *ACM Trans. Auton. Adapt. Syst.*, 4(2):14:1–14:42, May 2009.

[SVDP12] Alberto Sangiovanni-Vincentelli, Werner Damm, and Roberto Passerone. Taming Dr. Frankenstein: Contract-based design for cyber-physical systems. *European Journal of Control*, 18(3):217–238, 2012.

[SWG$^+$08] E.O. Schweitzer, D. Whitehead, A. Guzman, Y. Gong, and M. Donolo. Advanced real-time synchrophasor applications. In *Proc. of the 35th Annual Western Protective Relay Conference*, 2008.

[TAE15] Daniel Thiele, Philip Axer, and Rolf Ernst. Improving formal timing analysis of switched ethernet by exploiting FIFO scheduling. In *2015 52nd ACM/EDAC/IEEE Design Automation Conference (DAC)*, 2015.

[TAES14] Daniel Thiele, Philip Axer, Rolf Ernst, and Jan R. Seyler. Improving formal timing analysis of switched ethernet by exploiting traffic stream correlations. In *2014 International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS)*, 2014.

[TAP] TAPPS H2020 Project. Trusted Apps for open CPSs. http://www.tapps-project.eu/.

[TDA$^+$13] Daniel Thiele, Jonas Diemer, Philip Axer, Rolf Ernst, and Jan Seyler. Improved formal worst-case timing analysis of weighted round robin scheduling for Ethernet. In *2013 International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS)*, 2013.

[TE16a] Daniel Thiele and Rolf Ernst. Formal analysis based evaluation of software defined networking for time-sensitive ethernet. In *2016 Design, Automation Test in Europe Conference Exhibition (DATE)*, 2016.

[TE16b] Daniel Thiele and Rolf Ernst. Formal worst-case performance analysis of time-sensitive ethernet with frame preemption. In *2016 IEEE 21st International Conference on Emerging Technologies and Factory Automation (ETFA)*, 2016.

[TE16c] Daniel Thiele and Rolf Ernst. Formal worst-case timing analysis of ethernet TSNs burst-limiting shaper. In *2016 Design, Automation Test in Europe Conference Exhibition (DATE)*, 2016.

[TED15] Daniel Thiele, Rolf Ernst, and Jonas Diemer. Formal worst-case timing analysis of ethernet TSNs time-aware and peristaltic shapers. In *2015 IEEE Vehicular Networking Conference (VNC)*, 2015.

[TMKPF] J. Takalo-Mattila, J. Kiljander, F. Pramudianto, and E. Ferrera. Architecture for mixed criticality resource management in internet of things. In *Proceedings of 2014 TRON Symposium (TRONSHOW)*, pages 1–9.

[TMW$^+$17] Rohan Tabish, Renato Mancuso, Saud Wasly, Sujit S Phatak, Rodolfo Pellizzoni, and Marco Caccamo. A Reliable and Predictable Scratchpad-centric OS for Multi-core Embedded Systems. In *Real-Time and Embedded Technology and Applications Symposium (RTAS)*, pages 377–388. IEEE, 2017.

[TOG$^+$14] Salvador Trujillo, Roman Obermaisser, Kim Gruettner, Francisco Cazorla, and Jon Perez. European project cluster on mixed-criticality systems. In *Design, Automation and Test in Europe (DATE) Workshop 3PMCES*, 2014.

[TSAE15] Daniel Thiele, Johannes Schlatow, Philip Axer, and Rolf Ernst. Formal timing analysis of CAN-to-Ethernet gateway strategies in automotive networks. In *Real-Time Syst (2016) 52*, 2015.

[tsn12] Time-sensitive networking task group, 2012.

[TSN16] TSN. 802.1Qbu-2016 - IEEE standard for local and metropolitan area networks – bridges and bridged networks – amendment 26: Frame preemption, 2016.

[TSN17a] TSN. 802.1CB - frame replication and elimination for reliability, 2017.

[TSN17b] TSN. 802.1Qci - per-stream filtering and policing, 2017.

[TTDL12] Martin Tørngren, Stavros Tripakis, Patricia Derler, and Edward A. Lee. Design contracts for cyber-physical systems: Making timing assumptions explicit. Technical Report UCB/EECS-2012-191, EECS Department, University of California, Berkeley, August 2012.

[ubm13] 2013 - embedded market study. Technical report, UBM Tech, 2013.

[VMW] VMWare. A performance comparison of hypervisors. whitepaper. http://www.vmware.com/pdf/hypervisorperformance.pdf.

[VY15] P. K. Valsan and H. Yun. MEDUSA: A predictable and high-performance dram controller for multicore based embedded systems. In *2015 IEEE 3rd International Conference on Cyber-Physical Systems, Networks, and Applications*, pages 86–93, August 2015.

[WHR14] J. Whittle, J. Hutchinson, and M. Rouncefield. The state of practice in model-driven engineering. *IEEE Software*, 31(3):79–85, May 2014.

[WLS97] C.J. Walter, Patrick Lincoln, and Neeraj Suri. Formally verified on-line diagnosis. *IEEE Trans. Softw. Eng.*, 23(11):684–721, November 1997.

[Yu09] Y. Yu. *Modeling and reasoning timing constraints in open distributed real-time and embedded systems*. PhD thesis, Illinious Institute of Technology, 2009.

[YYP$^+$13a] H. Yun, G. Yao, R. Pellizzoni, M. Caccamo, and L. Sha. MemGuard: Memory bandwidth reservation system for efficient performance isolation in multi-core platforms. In *2013 IEEE 19th Real-Time and Embedded Technology and Applications Symposium (RTAS)*, pages 55–64, April 2013.

[YYP$^+$13b] Heechul Yun, Gang Yao, Rodolfo Pellizzoni, M. Caccamo, and Lui Sha. Memguard: Memory bandwidth reservation system for efficient performance isolation in multi-core platforms. In *19th IEEE Real-Time and Embedded Technology and Applications Symposium, RTAS 2013, Philadelphia, USA, April 9-11*, pages 55–64, 2013.

[ZBSL14] Michael Zimmer, David Broman, Chris Shaver, and Edward A Lee. Flexpret: A processor platform for mixed-criticality systems. In *2014 IEEE 19th Real-Time and Embedded Technology and Applications Symposium (RTAS)*, pages 101–110. IEEE, 2014.

[ZP13] Marc Zeller and Christian Prehofer. Modeling and efficient solving of extra-functional properties for adaptation in networked embedded real-time systems. *Journal of Systems Architecture*, 59(10):1067–1082, 2013.