Embedded multi-core systems for mixed criticality applications in dynamic and changeable real-time environments

Project Acronym:

EMC^2

Grant agreement no: 621429

Deliverable no. and title	**D1.1 – State of the art and SoA architecture requirements report**	
Work package	WP1	SP_SoA - Embedded system architecture
Task / Use Case	T1.1	Requirements and specification
Subtasks involved		
Lead contractor	Infineon Technologies AG Dr. Werner Weber, mailto:werner.weber@infineon.com	
Deliverable responsible	Thales Alenia Space Italy SPA (TASI) Lucia Amorosi, mailto:lucia.amorosi@thalesaleniaspace.com Dario Pascucci, mailto:dario.pascucci@thalesaleniaspace.com	
Version number	v1.1	
Date	24/10/2014	
Status	Final	
Dissemination level	Public (PU)	

Authors

Partici-pant no.	Part. short name	Author name	Chapter(s)
02D	TTT	Andreas Eckel	Contributions in chapters 1, 2, 3, 4 and 5
02F	VIF	Andrea Leitner	Contribution to 3.3 (system safety)
16D	LTU	Jonas Gustafsson	Added contributions to chapter 1 and updated reference list.
15L	HIB		General contribution
11J	TASI	L.Amorosi, D.Pascucci	General contribution
16A	Chalmers	Risat Pathan	Contribution in: State of Art
02A	AVL	Armengaud Eric	Contributions in: Related work and related projects
16B	Ericsson	Patrik Ekdahl	Contribution in State of Art (security)
15O	SevenS	Javier Díaz	General contribution
13A	ISEP	Luis Miguel Pinho	General contribution
02G	AIT	Weißenfeld Axel	General contribution

Document History

Version	Date	Author name	Reason
v0.1	13/07/2014	Alfred Hoess	Initial Template
v0.2	19/08/2014	Jonas Gustafsson	First suggestion on deliverable structure
v0.3	28/08/2014	Jonas Gustafsson	Added some initial text
v0.4	2014-10-07	Andreas Eckel	TTT contribution
v0.5	14/10/2014	Andrea Leitner	VIF contribution
v0.6	15/10/2014	L.Amorosi/D. Pascucci	TASI & HIB & LTU & Chalmers & AVL Contributions
v0.9	17/10/2014	Jonas Gustafsson	Minor changes
v1.0	18/10/2014	Alfred Hoess	Final editing
	21/10/2014	L.Amorosi/D. Pascucci	ISEP & AIT & SevenS Contributions
v1.1	24/10/2014	Alfred Hoess	Deliverable review, final editing and formatting, deliverable submission

Publishable Executive Summary

Within EMC^2 WP 1 focusses on Service oriented Architectures (SoA). Such approach has been investigated during numerous predecessor projects. The benefit and cross domain applicability of SoA has been demonstrated and ranges from reduction of development time and cost up to increased reliability due to re-using proven services. Combining them with new services in order to obtain new functionalities and even new applications in potentially other industrial domains is significantly more beneficial compared to “re-inventing the wheel” over and over again.

Today SoA also provides enhanced opportunities for use in many-core / multi-core applications. The work of EMC^2 in particular concerning this WP will focus on integrating suitable and appropriate means for many core/multicore based system designs.

Table of contents

List of figures

List of tables

1. Introduction

In this deliverable report we are presenting state of the art technology within Service Oriented Architecture (SOA) for embedded systems in mixed critical applications. The document has a general approach to the field of embedded systems architecture, to not exclude any specific “living lab” interest.

Notable for this report is that the time-frame for completing this deliverable has been very short, and a thorough analysis of all requirements still is to be done in relation to high-level project-wide requirements.

1.1 Objective and scope of the document

The objective of this document is to present the state of art of SoA, and how it can be applied on multi-core networked systems. The document also includes detailed requirements, specified by the project partners, which form the basis for the continued work within WP1 and throughout the project.

1.2 Structure of the deliverable report

Chapter 1: Introduction (this chapter)

Chapter 2: State of art
The state of art analysis of Service oriented Architecture, where technologies and standards commonly used in the industry today are briefly reviewed, and new standards that are expected to be of interest to the project are looked into.
Chapter 2 also includes references to related work and projects that are of special interest to this project. A short summary of highly relevant projects are to be found here.

Chapter 3: Requirements
This chapter highlights a selection of requirements from the complete list of requirements (Appendix 2), and motivates their importance in the context of system aspects.

Chapter 4: Discussion
Discussions about the question whether the requirements can be fulfilled by the state of the art or if there are large gaps.

Chapter 5: Conclusion

Chapter 6: References

Chapter 7: Appendix
7.1 Abbreviations
7.2 Excel file with technical specifications.

2. State of Art

The term Service-Oriented Architecture (SOA) has gained a lot of interest from both the academic world as well as from the industry. SOA is an architectural design pattern for distributed functionally with loosely coupled interfaces [1]. A service is the core building block of SOA, and it constitutes a software block designed to address a certain task. A service must have with a formal interface described using a standardized description framework (shown in Fig. 1). For Web services, Web Service Description Language (WSDL) and Web Application Description Language (WADL) are often used.

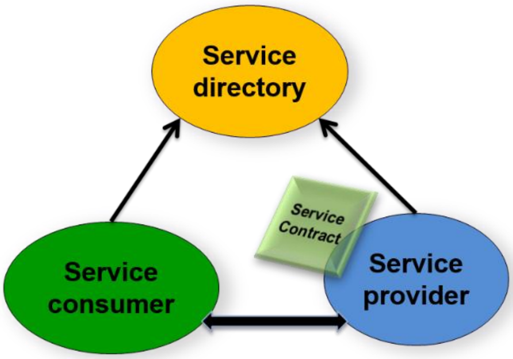

Fig. 1. Service-Oriented Architecture overview

A service must meet certain requirements and properties, such as being discoverable, composable and loosely coupled from any operating system, programming language and other services. Normally, the use of SOAs is driven by certain properties and requirements, as discussed by Erl in [1]. Some of the most important trade-offs when comparing SOA to legacy client-server models for distributed systems are, as stated in the SOA manifesto:

- Business value over technical strategy;
- Intrinsic interoperability over custom integration;
- Flexibility over optimization.

The use of SOA is driven by a need for re-use of code and systems over the use of specialized building blocks, optimized for a certain application or scenario. A service should hide any internal logic, which could be implemented using any (suitable) programming language on any (suitable) operating system. A service should be loosely coupled, with no predefined connections. Existing services can also be composed into new services using either *Service Orchestration* or *Service Choreography*.

However, even if the use of SOA is a very powerful development paradigm, the use of a potentially large number of functionalities distributed over many different systems, over different protocols and implemented in several programming languages imposes certain challenges when it comes to document how services and systems should be described and their interactions maintained.

Service Orchestration

Service Orchestration is the simplest type of service composition where one service or system acts as a coordinator between all services involved. This approach is relatively easy to set up and monitor since all information is available at a central entity.

Service Choreography

In Service Choreography, there is no central service or system involved. Instead, all composed services work independently with each other in a completely distributed manner.

2.1 Service Oriented Architecture

Interoperability by exposing functionality as services has been in development since the 1990s with early popular Microsoft technologies such as OLE, COM/DCOM, and OPC characterized by their proprietary, platform-specific, and non-extensible nature. With the emergence of the Internet and open W3C standards such as XML, SOAP, XSD, and WSDL, the technologies required making truly interoperable, cross-platform, and extensible services became available to everyone. Recognizing the importance of these characteristics, many if not all of the subsequent service technologies utilize either the WS-* standards, or the design principles of the web such as REST.

Besides this trend, another emerging trend relevant to the context of embedded systems is the emergence of technologies to allow resource-constrained devices to expose their data as (web) services, for improved integration with enterprise systems (e.g. DPWS, CoAP). These applications of SoA technologies in embedded devices are especially promising in the area of (wireless) sensor networks, where they can deal with the heterogeneous character of these systems to lower complexity and decrease development, deployment and maintenance costs.

2.1.1 Software design using SoA

Service-Oriented Architecture (SoA) is a software design methodology in which systems are seen a set of separate software components that provide services to other applications. Henceforth, larger applications can combine services with other software applications in order to provide a more powerful functionality, without having to build everything from scratch.

A main concern with building new applications that make use of SoA frameworks is that of trusting that the services provide the necessary or expected functionality. That is, some service may be expected to produce some certain pattern of output, but due to some faulty implementation or misinterpretation of the requirements, it may be providing actual inconsistent or erroneous data. As low-cost embedded software grows in the way it interacts with our quotidian, this problem inherent with the development of services may lead to undesirable consequences.

One way to address this problem is to rely on the principles of Design-by-Contract (DbC), popularized by Meyers and the Eiffel [2] programming language, with formal background that is being intensively studied and improved in the area of Formal Software Verification (FSV) since it first appeared in the works of Floyd and Hoare in the 60's [3]. In DbC, a formal agreement between the component produced and the consumer is established, normally in the form of a specification language that has expressive power (and reasoning mechanisms) to capture the properties of interest that must be established in the contract. The producer can use the specified contract and some FSV tool to formally prove the software that they are writing satisfies the formal content of the contract. On the other hand, the consumer can trust in the service of the producer, and use the contract in the verification of the software that will use that service.

DbC is mainly used in actual program verification of code, like in the case of the Eiffel [2] and Ada 2012 [4] programming languages, or the Spark [5] and Frama-C [6] verification frameworks. At the model level development, UML-like dialects are the de-facto technology taken into consideration by the majority of software development teams. For further enriching the model, it is customary to add informal descriptions to the model components, while the OCL [7] language and associated verification tools are usually not taken into consideration. However, exposing the modelling languages to the expressivity of contracts allows improving not only the understanding of the model, but also helps in guiding the functionality of coupled model-transformation and/or code-generation methods. Furthermore, contracts at the model level can also be used to improve the testing efforts by imposing constraints over the search-space, thus narrowing it and allowing for faster results or larger levels of coverage.

So far, DbC has been mainly targeted for functional correctness of programs. The results have been promising, and the number of algorithms and other software pieces that has been verified has grown considerably, namely when the usual supporting tools such as Model Checkers or Automated Theorem Provers started to become very efficient in proving properties that are common to programming. In the context of SoA, the functional correctness of programs is fundamental to ensure that a service complies with its contract, but in the case of some embedded systems, there is a natural necessity of having ensured some non-functional properties, for instance, like real-time properties.

To address non-functional properties, many of which are hardware and/or environment dependent, several formal timed languages and timed state machines have been proposed, by the research community, and implemented in successful tools like Uppaal [8] or KeY [9], and were used with success in the verification of the real-time and Cyber-Physical Systems (CPS). Although current functional correctness contracts verification and non-functional contracts verification technologies can be, at least up to some extent, integrated in a single SoA model-driven framework, there is no common supporting programming language that allows to express, in the form of a program, the interdependences and relations between the functional and non-functional properties of the system, thus forcing the intervention of programmers to complete the implementation.

2.1.2 Summary of SOA-standards relevant to the EMC^2-project

Today there are a lot of existing SOA-standards available for use, and a number of up-and-coming standards that are specifically targeting (resource constrained) embedded devices. Within the following sections we provide a summary of service oriented standards considered relevant to the project. These standards are used in the industry today or are gaining interest from developers in the embedded system architecture community. They are organized under the organization that is responsible for the standard.

2.1.2.1 W3C Web Services (WS-*)

SOAP

SOAP is a protocol for exchanging structured information between web services, and is maintained by the W3C Web Services Activity [10]. Messages contain one element (the envelope), with a header and a body as child elements. The format of the header and body contents is not dictated by the protocol. The message format of SOAP relies on XML Information Set (XML Infoset), which is a W3C specification to describe an abstract data model in terms of information items. As a consequence, although SOAP messages are typically represented in XML, the messages can also be represented as CSV or JSON. A major advantage of SOAP as a transport method for services over techniques such as DCOM is that SOAP is typically sent over HTTP channels, making them firewall friendly.

Web Services Description Language (WSDL)

While the SOAP protocol only defines the encapsulation of messages, the XML-based Web Services Description Language (WSDL) describes a web service as a combination of an abstract interface and a collection of bindings and endpoints on which the service can be contacted [11]. With a WSDL document, client code to access the service can be automatically generated for many different programming environments, or services can be dynamically invoked by parsing the specification at run-time.

Universal Description, Discovery, and Integration (UDDI)

With the combination of SOAP and WSDL standards, one can build a service accessible via the web by many different platforms in a standardized way. What the UDDI standard aims to add is a standardized way of finding available services by forming a ‘phonebook’ of web services [12].

2.1.2.2 Windows Communication Foundation (WCF)

The Microsoft Windows Communication Foundation (WCF), previously known under the codename ”Indigo” is a framework to build service-oriented applications and was first released with the .NET 3.0 framework in 2006 [13].

WCF is a unified API for message-based communication across different transport protocols such as HTTP, TCP, or Microsoft Message Queuing (MSMQ). Communication is done between so-called endpoints, each of which consists of an address, binding, and contract. The address defines where to find the service, the binding defines how to communicate with the service (transport protocol, message encoding, security requirements), and the contract defines the functionality of the service.

2.1.2.3 Open Platform Communications (OPC)

Originally developed in 1996 under the title Object Linking and Embedding (OLE) for Process Control, the OPC standard(s) are used for applications in Process Control, discrete manufacturing, building automation, and many others.

The OPC standard is currently developed and maintained by the non-profit OPC Foundation [14], which offers (paid) memberships. Although no company owns the protocol, and anyone is free to develop OPC servers/clients, not all specifications are open to non-members of the OPC foundation. Because of this, non-members may not always be able to adhere to the latest version of the standard.

OPC Classic
The original version of OPC was developed in a time where the Microsoft Windows platform dominated the industrial automation world. Because of this, OPC Classic is based on the Microsoft OLE and COM/DCOM technologies, and is limited to the Microsoft Windows platform.

OPC Unified Architecture (UA)
To address the needs of OPC users and to prepare OPC for the future, the OPC Unified Architecture (OPC UA) was developed and released in 2008 [15] [16] [17]. OPC UA is a service-oriented architecture that integrates all of the OPC Classic specifications (DA, A&E, HDA) into a single extensible framework. The main goal of OPC UA is to keep all of the functionality provided by OPC Classic, while replacing COM/DCOM with a SOA based on the open web services standards such as TCP/IP, HTTP, SOAP, and XML. With the move from COM/DCOM to web services, OPC UA brings platform independence, and can be implemented on platforms ranging from an embedded micro-controller to a cloud-based infrastructure.

2.1.2.4 OASIS

The Organization for the Advancement of Structured Information Standards (OASIS) is a non-profit consortium, originally founded under the name "SGML Open" in 1993. The mission of OASIS is as follows: *"To drive the development, convergence and adoption of market-driven open standards as building blocks for the global information society"* [18].

In the context of the ARROWHEAD project, the most relevant standard provided by OASIS is the Devices Profile for Web Services (DPWS) specification, which builds upon a collection of WS * specifications to enable embedded devices with constrained resources to expose their functionality through web services. For service orchestration in the context of business processes, the OASIS specification Business Process Execution Language for Web Services (WS-BPEL) provides a way to formally specify the choreography of (web) services, to form a business process [19] [20]. A WS-BPEL process can in turn be exposed as a web service and be utilized in another business process.

Devices Profile for Web Services (DPWS)

The Devices Profile for Web Services (DPWS) is an OASIS standard [21] to enable the implementation of secure web services on resource-constrained devices such as ThreadX, Windows CE, and VxWorks devices [22]. Because DPWS is fully aligned with the Web Services standards (Section 2.1.2.1), it has many of the advantages of the WS-* standards such as programming language and platform independence, scalability, security, and a high market acceptance. An early implementation was made in 2005 as part of the SIRENA project [23] [24]. DPWS has also been shown to be implementable even on highly resource-constrained devices, albeit with a limited subset and performance limitations [25].

2.1.2.5 Constrained Application Protocol (CoAP)

The Constrained Application Protocol (CoAP) is a specialized web transfer protocol for use with constrained nodes and constrained networks, designed for machine-to-machine (M2M) applications [26]. CoAP provides a request/response interaction model between application endpoints, supports built-in discovery of services and resources and includes key concepts of the Web such as URIs and Internet media types.

CoAP is a RESTful protocol implemented using asynchronous UDP messages with optional reliability. The protocol uses the verbs GET, PUT, POST, and DELETE which have a very similar meaning to the same HTTP verbs. Service discovery is supported in CoAP using either unicast to a known address/hostname, or multicast, in combination with the CoRE Link Format [27] to find service URIs.
The Constrained RESTful Environment (CoRE) working group of the Internet Engineering Task Force (IETF) has done most of the standardization work on CoAP.

2.1.3 Realtime aspects

There have been different works on real-time scheduling of mixed criticality systems. Several international projects like PROARTIS (www.proartis-project.eu [28]), PROXIMA (www.proxima-project.eu. [29]) and MCC (www.cs.york.ac.uk/research/research-groups/rts/mcc/) have addressed scheduling aspects for mixed-criticality systems. These works can broadly be categorized in two directions: (i) certification-cognizant scheduling of mixed-criticality systems, and (ii) performance-oriented scheduling of mixed-criticality systems.

The main issue that the former category, i.e. certification cognizant scheduling, addresses is the possibility of “execution-time overrun” of some low-critical tasks, which can hinder the execution of high-critical tasks and can cause to miss their real-time deadlines. The solution is to detect such overrun event and then to adapt the real-time schedule to correctly dispatch high-critical tasks to meet their deadlines. Note that low-critical tasks may miss their deadlines due to such adaptation of the schedule. However, this adaptation of the schedule can prove to the certification authority (who approves the final product) that the system’s high-critical tasks always meet all deadlines even if some low-critical tasks are behaving abnormally.

On the other hand, the main issue that the latter category, i.e. performance-oriented scheduling, addresses is how computing resources can be partitioned among the tasks in different criticality levels such that tasks of one criticality level do not interfere with the tasks in another criticality level across their resource partitions. The aim in this respect is to ensure that resources given to the high-critical tasks are never compromised by abnormal behavior of low-critical tasks. Moreover, low critical tasks have their own (possibly limited) share of computing resource which guarantees the QoS of those tasks. A recent survey that discusses the different works (up to June 2014) on mixed-critical systems can be found in [30].

The main aim of task T1.4 in EMC^2 project is to consider other aspects (not only the execution overrun problem) which can be used to define and adapt the real-time schedule of mixed-criticality systems. For example, a recent work [31] considers “frequency of faults” over a time frame to detect a situation where recovering from such faults in low-critical tasks may cause high-critical tasks to miss their deadlines. To avoid missed deadlines, only high-critical tasks are scheduled whenever such an event is detected. Due to the open nature of “interacting components” of mixed-critical systems that EMC^2 project considers, different run-time situations can cause high-critical tasks to miss their real-time deadlines. The aim of task T1.4 is to identify different types of such run-time events (subtask ST1.4.1) and to take appropriate real-time scheduling decisions so that the deadline of the high critical tasks are always met while the performance of the low-critical tasks are minimally compromised (subtask ST1.4.2).

2.1.4 Service Oriented Architecture Security

SoA was initially used mainly in a web context using HTTPS in conjunction to technologies such as SOAP and REST with XML or JSON packaging. The HTTPS protocol utilizes the SSL or TLS cryptographic protocols for secure transfer of data. Access control is sometimes based on technologies such as XACML (the extensible access control markup language) [32].

From a security perspective, it is desirable to have three basic properties fulfilled when utilizing web services: Confidentiality, Integrity, and Authentication. The authentication is a service policy mechanism. It is needed to supply the correct level of service to different users or classes of users (e.g. the freemium business model). In the case of premium customers only, the authentication provides some resistance to denial of service attacks by verifying the user at an early stage before too many resources are spent. The confidentiality and integrity is today mostly solved by some external tunnel, for example securing the client to host communication by HTTPS/TLS. This does not provide the desired message or object security but has become modus operandi due to the widespread deployment of HTTPS/TLS. Some frameworks, such as the Microsoft .NET WCF environment has come further by implementing message security in its SOA classes, but this requires client support as well and is not as interoperable as HTTPS/TLS in today’s web architecture.

From an attack perspective it is also important to consider the fact that SOAP/XML is a very powerful protocol. SOAP can contain external references that compliant services need to “dereference”, i.e., download. This has implications similar to cross-scripting attacks in HTML. Furthermore, XML can contain resource URIs to local (server side) files, and information from or about that file may be leaked. From a security perspective, a more restricted protocol would be desirable.

The same, or even more, holds for XACML. Here we have already activities in IETF that strive to find less complex solutions for the IoT and these would be beneficial for embedded systems tool. A more in depth overview what is achieved will be part of the state-of-the-art study.

Because we are interested in addressing how components autonomic driven by policy of security SLAs can interact and configure themselves there must be a SOA framework that supports this. There are today such frameworks and some have even a security architecture, e.g. IBM’s SOA framework. However as initially said these are highly web centric and do not sufficiently address the problem how components establish trust in the environment they have to function.

Typically these frameworks focus on what enterprise architects care about, such as:

- The need for identity to be decoupled from the services. All entities in SOA have identities - users, services, and so on, that needs to be properly identified so that appropriate security controls can be applied.
- The need to seamlessly connect to other organizations/components on a real-time, transactional basis
- The need to ensure that, for composite applications, proper security controls are enacted for each service and when services are used in combination
- The need to manage identity and security across a range of systems and services that are implemented in a diverse mix of new and old technologies

- Protection of business data in transit and at rest
- The need for demonstrable compliance with a growing set of corporate, industry, and regulatory standards

These are important by themselves but the realization depends at large that the components do have (certain) trust in the processing performed in other components which goes beyond the identity and access control management perspective. Here the state-of-the-art study should identify frameworks that meet the general usage requirements in the EMC^2 project and that have functionalities that can be used as a basis for long-life management of trust functions.

Service oriented architecture has also become available in the domain of embedded systems. However, given that embedded devices are often resources constrained, often other technologies and protocols have been utilized to achieve SoA. Notable examples are the Constrained Application Protocol (CoAP) [26], and the UDP-capable DTLS (Datagram Transport Layer Security) that replaces TLS [33].

Due to their low-power nature, embedded devices are susceptible to a wide range of denial of services attacks. Therefore some computation intensive technologies such as XML and XACML are often replaced with simpler counterparts such as the Efficient XML Interchange format [34] and access control technologies such as technologies such as oAuth [35]. Another interesting approach for simplifying authorization is to off-load some tasks to more powerful nodes in the network, see for example [36].

It should be noted that use of wireless or ad hoc networks in many embedded systems creates a number of relevant security challenges. Some recent relevant attacks include sophisticated jamming attacks [37] and attacks against the network routing model [38].

2.1.5 System Safety

Most embedded systems considered in EMC² are used in a safety-critical environment. In many domains rigorous functional safety standards need to be applied, because malfunctions of E/E systems potentially cause hazardous events e.g. in the automotive domain for driver, passengers, other traffic participants, and uninvolved third party. Functional safety aims to get a clear understanding of potential problems, their causes and effects, and provides possible fault solutions and mitigation measures to ensure a safe state in every hazardous situation.

There are various safety standards applicable in different domains, e.g. the generic safety standard IEC 61508, the automotive standard ISO 26262, or DO-178B in the aerospace domain. Current safety standards provide guidance on how systems can be developed to be safe. Safety standards however suffer from a few issues that need to be addressed to meet the needs of future CPS, such as increasingly automated transport and mobility systems. One main concern is the lack of support for dynamically evolving systems and guidance for reusing components and subsystems in a certifiable way. This could potentially contradict the SOA concept or “Safety as a Service” could be one approach with good prospects of success.

In order to achieve a safe embedded system, safety needs to be considered already during development and has to be included in several aspects: analysis of architectural design, inter-core communication & synchronization, memory usage, managing access to shared resources such as cache, etc.

Safety for multi-core systems in general:

Lockstep systems are fault-tolerant computer systems that run the same set of operations at the same time in parallel on different cores. This redundancy enables error detection and error correction. The output of the redundant cores can be compared in order to determine if there has been a fault. This requires at least two systems (dual modular redundancy). An error can be automatically corrected if there are at least three systems (triple modular redundancy).

Multi-core architectures can also be used to provide parallel but diverse realizations of an algorithm on different cores. This can be used for ASIL (Automotive Safety Integrity Level) decomposition in automotive. An element implemented to address a given safety goal, with a given ASIL may be decomposed into two independent elements, with possibly lower ASIL. This requires that the cores are independent and cannot interfere with each other - Freedom from Interference. Real freedom of interference is only possible if there exists some kind of Memory Protection Unit, which makes sure that applications can only access previously defined memory spaces.

Another existing technique is that the OS or a dedicated unit can be used to monitor the program flow. This means that it should check whether tasks have an exceeding runtime or block interrupts longer than appropriate. There are existing approaches based on checkpoints which support this measure.

Considerations for mixed-criticality:

Safety-related components require temporal and spatial separation from other system components of different levels of criticality. Today’s separation concepts (at least in automotive) are mostly designed to use completely independent subsystems for each function. Usually, there are separate control units for safety-related aspect and for human machine interaction.

Multi-core processors allow devices to be partitioned so that specific functions can be performed on dedicated cores. This ensures high performance while offering the ability to segregate functions. Segregation means the separation or isolation of safety-related functions from general-purpose functions of a device. Without segregation, everything would need to be certified to the highest integrity level. Thus, the amount of code that needs to be certified is significantly reduced, which lowers product costs while increasing time-to-market. Another opportunity that arises from this segregation is the ability to update or enhance the general-purpose partitions without modifying the safety-critical applications.

Summarized this means that there are already a lot of techniques available which could be used in mixed-critical systems. Nevertheless, currently missing are detailed investigations of safety requirements for service-oriented architecture and the selection of appropriate measures to lower risks to an acceptable level. And recently a SOA safety concept should be closely linked with security (e.g. EURO-MILS FP7).

System service criticality capabilities

- Real time capabilities
- Energy usage and computation efficiency
- System Power Supply efficiency and flexibility
- AMSPS functionality and safety
- System robustness and fault tolerance
- Performance predictability
- Variability adaptability

2.2 Related work and related projects

2.2.1 Arrowhead (Artemis)

Short summary

Arrowhead’s vision is to enable collaborative automation by networked embedded devices. Arrowhead assumes that a service-based approach will be the feasible technology that enables collaborative automation in an open-network environment connecting many embedded devices. The multi-billion device/service perspective places a very strong demand on the interoperability and integrability of devices and services provided by the multitude of players in the market place. The major technology achievements will the creation of the Arrowhead interoperability framework – A technology framework enabling collaborative automation and closing innovation critical technology gaps.

Thus, Arrowhead grand challenges are:

- Enabling the interoperability of services provided by almost any device, and
- Enabling the integrability of services provided by almost any device.

Relation to EMC^2

By utilizing the concepts developed in the Arrowhead project, EMC^2 will have a solid ground for continued research for the system architecture objectives with the special requirements defined in the EMC^2 project.

2.2.2 IMC-Aesop (FP7)

Short summary

Several European projects have been initiated that attempt to create a service-based model for the control of process control systems. One of the most notable is the IMC-AESOP project (http://www.imc-aesop.eu/) that investigates a Service Oriented Architecture approach for monitoring and control of very large scale Process Control Systems.

The IMC-AESOP project goal is to show the feasibility of a SOA-based approach for the very large systems that are generally used in production plants. The IMC-AESOP project comes forth from other EU-funded projects that focus around SOA within the industrial automation industry.

The IMC-AESOP project builds upon the results of other EU projects such as SIRENA (https://itea3.org/project/sirena.html) and SOCRADES (www.socrades.eu) [39]. These two projects implemented and evaluated the feasibility of replacing the typical automation pyramid protocols with SOA based protocols. The focus in these projects was around the Device Profile for Web Services (DPWS) technology stack. The IMC-AESOP project on the other hand used a variety of different protocols, amongst them DPWS [21] and OPC-UA [17].

Relation to EMC^2

Some concepts and strategies initiated by the IMC-AESOP project have matured and are further developed in other projects such as Arrowhead. Some of the SOA-based concepts can also be of relevance for the EMC^2 project.

2.2.3 GENESYS (FP7)

Short summary

The Genesys project defines an architecture candidate for embedded systems within ARTEMIS. The architecture provides a suitable template for designing embedded systems, independent from the application domain. It defines systems on three different abstraction levels: chip, device, system (open or closed). The components are connected via specialized interfaces such that properties of a component as a computational unit are preserved, and also faults and errors are contained. The service structure is also ordered in three levels: basic services provided independent of an application domain, optional services that can be implemented if needed by a specific application, and strict application services unique for a single application.

Relation to EMC^2

EMC^2 will endeavor to enhance the results from GENESYS, INDEXYS and ACROSS in the direction of improving the performance of the developments made. Thus the initial basis for the related developments and demonstrations is laid down in the GENESYS project. It defines a blue print reference architecture for embedded systems design. This can be understood similar to a template used in generating a document just simply for using such blue print/template architecture to design an embedded system.

It is a first ARTEMIS candidate for a universal Service-oriented Architecture (SoA). The concept has been evaluated on two important European projects: INDEXSYS, and ACROSS. These projects provided highly conformational results. Adopting the same concepts in EMC^2 will provide excellent basis for future SoA proposals.

2.2.4 INDEXYS (Artemis)

Short summary

Directly following the results produced in GENESYS, INDEXYS intended to demonstrate that the GENESYS results can be applied to cross-industrial applications and will in addition bring benefits to the development engineer designing an embedded system based on the GENESYS architectural blue print. It provided clear evidence that SoA approach defined in GENESYS offers significant benefit such as reductions in development time and cost, reduced integration effort, re-use of hardware and software building blocks for embedded systems, a common architectural approach enabling to rely designs on already proven repository archived components (hardware and software) and several other advantages to use one and the same architectural approach. INDEXYS also proofed that such benefit can be shown for aerospace-, automotive- and railway industrial domain and thus provides evidence that also other industrial domains would benefit in a similar way and manner as demonstrated for the three demonstrator domains. It closely cooperated with ACROSS and used also results from ACROSS in order to increase the confidence level for demonstrating the cross domain benefit (in addition to the three small size demonstrators in INDEXYS the five large scale demonstrators from ACROSS were included in the evaluations for INDEXYS).

Relation to EMC^2

Alike GENESYS, INDEXYS was a forerunner project to evaluate and demonstrate the advantages of SoA and the architectural blue print approach as defined in GENESYS. Thus INDEXYS served as an enabler for follow-up projects like ACROSS or now EMC^2. Its results were necessary to allow argumentation in the direction of claiming to fill the niche of blue print architectural templates when promoting the GENESYS results as unique approach for embedded systems design.

2.2.5 ACROSS (Artemis)

Short summary

Following the INDEXYS project ACROSS provided industrialization of the ideas developed in GENESYS and INDEXYS by developing a Multi-Processor-System on a Chip MPSoC solution, a dedicated tool set to configure the new devices and produce suitable scheduling and appropriate middleware.

A cross-domain architecture for MPSoC (ACROSS) represents a unified conceptual and practical solution for mixed-criticality systems independent from the industrial domain. The architecture is based on MPSoC (SoC) with deterministic interconnect called Time-Triggered Network-on-Chip (TTNoC) among different cores. Communication between TTNoC and cores is performed over deterministic linking interfaces which enable full spatial and temporal isolation; furthermore they serve as a fault and error containment perimeter. It makes use of a dedicated system of fragment switches allowing connecting cores in a very special and reliable manner implementing the time- and space isolation. In addition, ACROSS design allows combination of heterogeneous cores on a single chip. The ACROSS was evaluated on five industrial demonstrators: hybrid vehicle hardware in the loop (HIL), visual landing aid system, cooperative Unmanned Aerial Vehicle (UAV) system, industrial control, and mobile communications.

Relation to EMC^2

The ACROSS project is a result of a long research effort of European industrial and research communities. It is a direct realization of the Service-oriented architecture (SoA) for mixed-criticality systems developed in the predecessor project GENESYS. It provides unique approach to solve problems of concurrency and determinism, necessary for mixed-criticality applications. In essence, it is highly compatible with objectives of EMC^2. The ACROSS prototype can be observed as a conceptual and practical template for a new enhanced platform designed within EMC^2.

2.2.6 EURO-MILS FP7

EURO-MILS is an ongoing FP7 project. The mission of the EURO-MILS project is to develop a solution for virtualization of heterogeneous resources and to provide strong guarantees for isolation of resources by means of Common Criteria certification with usage of formal methods. MILS is a high-assurance security architecture that supports the coexistence of untrusted and trusted components, based on verifiable separation mechanisms and controlled information flow.

Relation to EMC^2

One challenge for a safety and fault tolerant concept for SOA is the link between safety and security. EURO-MILS address the security topic and the approach has been tried and tested. Based on this experience, the outcome can be used as a reference for security architectures.

2.2.7 CESAR (Artemis)

Link: http://www.cesarproject.eu/

Short summary:

"CESAR" stands for Cost-efficient methods and processes for safety relevant embedded systems and is a European funded project from ARTEMIS JOINT UNDERTAKING (JU).

The three transportation domains automotive, aerospace and rail, as well as the automation domain share the need to develop ultra-reliable embedded systems to meet societal demands for increased mobility and ensuring safety in a highly competitive global market. CESAR has provided significant and conclusive innovations in the two most improvable systems engineering disciplines:

- Requirements engineering in particular through formalization of multi viewpoint, multi criteria and multi-level requirements,
- Component based engineering applied to design space exploration comprising multi-view, multi-criteria and multi-level architecture trade-offs.

To maintain the European leading edge position in the transportation as well as automation market, CESAR aims to boost cost efficiency of embedded systems development and safety and certification processes by an order of magnitude. CESAR pursuits a multi-domain approach integrating large enterprises, suppliers, SMEs and vendors of cross sectorial domains and cooperating with leading research organizations and innovative SMEs. The CESAR project started in March 2009 and has successfully been finished by end of June 2012 after 40 months of project duration.

Relation to EMC^2:

There are at least two important interactions with EMC². The first one is related to the methods (requirements engineering, component-based engineering, safety engineering) required for the development of highly dependable systems. Even if CESAR was focused more on the system level and EMC² more on the component level, a good integration of the developed methods is definitely required. The second aspect is related to tool integration and interoperability. Mature concepts with different demonstrators have been developed within the CESAR project. This interoperability specification has been handed over to Artemis' MBAT and CRYSTAL projects, and shall be fully in line with the work performed in EMC² WP5 "system model design, tool platform and interoperability".

2.2.8 MBAT (Artemis)

Link: https://www.mbat-artemis.eu/

Short summary:

One of the most important strategic sectors in which Europe is developing, integrating and delivering high-quality products is the transportation domain. Here, high-class safety-related products as e.g. airplanes, cars and trains have a huge market impact. More and more of the market value of these vehicles is gained by embedded systems inside these products, and the number and importance of these embedded systems is steadily growing. One of the most important enablers to assure the quality of embedded systems is the application of powerful validation and verification (V&V) technologies accompanying the embedded systems development process. Unfortunately, the V&V technologies already in industrial use are still too expensive while often not effective enough.

ARTEMIS project MBAT will provide European industry with a new leading-edge V&V technology in form of a Reference Technology Platform (MBAT RTP) that will enable the production of high-quality and safe embedded systems at reduced cost in terms of time and money. This will be made possible by a new and very promising approach in which model-based testing technologies will be combined with static analysis techniques. Besides this combination, a further new approach is to use (and re-use) specially designed test & analysis models as basis for model-based V&V. This advanced model-based V&V technology will lead to a more effective and at the same time cost-reducing approach compared to traditional ones. In addition, MBAT RTP will be connected to other ARTEMIS RTPs to extend existing platforms. Developed by industrial key players (large companies and SMEs) in this domain and supported by leading research partners, the MBAT RTP will be of high value for the European industry, providing very effective means to assure utmost quality embedded systems at reduced costs.

Relation to EMC^2:

Similar to CESAR, the ARTEMIS MBAT project provides at least two major interactions with EMC². The first one is related to the methods for analysis, validation and test of the system. The methods proposed in MBAT (at system level) shall be in line with the one proposed in EMC² (at sub-system and component level). Hence, consistent and traceable validation from the components to the system is mandatory for the development of highly dependable (reliable, safe and secure) systems. The second aspect is the work performed in the context of interoperability specification and integrated tool chain. This work has a strong impact to EMC² WP5 “system model design, tool platform and interoperability”.

2.2.9 CRYSTAL (Artemis)

Link: http://www.crystal-artemis.eu/

Short summary:

The ARTEMIS Joint Undertaking project CRYSTAL (CRitical sYSTem engineering AcceLeration) takes up the challenge to establish and push forward an Interoperability Specification (IOS) and a Reference Technology Platform (RTP) as a European standard for safety-critical systems.

This standard will allow loosely coupled tools to share and interlink their data based on standardized and open Web technologies that enable common interoperability among various life cycle domains. This reduces the complexity of the entire integration process significantly. Compared to many other research projects, CRYSTAL is strongly industry-oriented and will provide ready-to-use integrated tool chains having a mature technology-readiness-level (up to TRL 7).

In order to reach this goal, CRYSTAL is driven by real-world industrial use cases from the automotive, aerospace, rail and health sector and builds on the results of successful predecessor projects like CEASAR, SAFE, iFEST, MBAT on European and national level.

Creating and establishing a new standard on a large scale in an already consolidated market cannot be achieved by individual organizations. With a budget of more than 82 million Euro and 71 partners from 10 different European countries, CRYSTAL has the critical mass to accomplish this endeavor. The project consortium is made up of participants from all relevant stakeholders, including OEMs, suppliers, tool vendors and academia.

Relation to EMC^2:

Main focus of the ARTEMIS CRYSTAL project is set on Interoperability Specification (IOS) and Reference Technology Platform (RTP) – with the target of achieving a higher maturity level (TRL - technology-readiness-level). This work and the respective results need to be well synchronized with EMC² - especially WP5 – in order to provide consistent outcomes at European level between the different research programs.

2.2.10 VETESS (Artemis)

Link: http://vetess.eu/

Short summary:

VeTeSS (Verification and Testing to Support Functional Safety Standards) is an ARTEMIS project to develop standardized tools and methods for the verification of safety properties, particularly for generic components and subsystems used in safety-relevant embedded systems. The project will concentrate on the strategically important automotive market in order to constrain the scope of the problem, limit the size of the consortium and the budget and to keep the project focused.

We are aware that many of the same problems arise in other sectors, and the project will actively engage, through an Industrial Advisory Board (IAB) created as part of the project, with representatives of those industries to share knowledge and disseminate results.

VETESS is a 36 month project, starting in May 2012, with Infineon UK as the lead partner.

Relation to EMC^2:

The ARTEMIS VETESS project is focused on the automotive domain and the efficient application of ISO26262. This project mainly impacts WP6 (system qualification and certification) and WP7 (automotive domain) to provide a basis how the different safety aspects (from management, system, SW, HW, calibration point of view) shall be integrated. It is expected that the other EMC² technical work-packages are impacted as well – at least for processes, methods, tools and standard related to the automotive domain (e.g., AUTOSAR).

2.2.11 SafeCer (Artemis)

Link: http://safecer.eu/

Short summary:

SafeCer (Safety Certification of Software-Intensive Systems with Reusable Components) is targeting increased efficiency and reduced time-to-market by composable safety certification of safety-relevant embedded systems. The industrial domains targeted are within automotive and construction equipment, avionics and rail. SafeCer will also develop certification guidelines and a training example for other domains, thus considerably increasing its market impact.

A primary objective is to provide support for system safety arguments based on arguments and properties of system components as well as to provide support for generation of corresponding evidence in a similar compositional way. By providing support for efficient reuse of certification and stronger links between certification and development, component reuse will be facilitated, and by providing support for reuse across domains the amount of components available for reuse will increase dramatically. The resulting efficiency and reduced time to market will, together with increased quality and reduced risk, increase competitiveness and pave the way for a cross-domain market for software components qualified for certification.

SafeCer brings together leading companies and SMEs across Europe (including OEMs, technology, tool, and competence providers, as well as certification and standardization experts), which together with selected universities and research institutes are capable and motivated to realize the SafeCer objectives.

The project is coordinated by Volvo Technology Corporation.

Relation to EMC^2:

The topic of composable safety certification is a key aspect of EMC² WP6 (system qualification and certification) in order to break down the complexity of system certification into the certification of single components and the capability to integrate these components-related certifications. The outcomes of SafeCer will be integrated as basis for WP6 to take into account the main project achievements for future developments.

2.2.12 RECOMP (ARTEMIS)

Short summary:

RECOMP (Reduced Certification Costs Using Trusted Multi-core Platforms) was a European funded project from ARTEMIS JOINT UNDERTAKING (JU) that started April 1st, 2010 and had a duration of 36 months. RECOMP research project contributed to the European Standard Reference Technology Platform by enabling cost-efficient certification and re-certification of safety-critical systems and mixed-criticality systems. The aim was establish methods, tools and platforms for enabling cost-efficient (re-) certification of safety-critical and mixed-criticality systems. Applications addressed were automotive, aerospace, industrial control systems, and lifts and transportation systems.

RECOMP provided reference designs and platform architectures, together with design methods and tools, for achieving cost-effective certification and re-certification of mixed-criticality, component based, multicore systems.

Relation to EMC^2:

RECOMP platform methods provide relevant inputs to EMC^2 project. Remarkably, the inputs related with shared resources utilization on multicore devices are extremely useful. The mechanisms provided by RECOMP formally establish a methodology capable to take advantage of the resources in use by multicore and many-cores devices, integrating software, hardware and development tools in a unified perspective. These inputs are significantly useful for the EMC^2 developments and Living Labs.

3. Requirements

3.1 Requirement structure

The requirements in this deliverable are of technical nature and are structured under system aspects provided including descriptions in the Appendix 2 of this document. In order to demonstrate the project wide approach of the EMC^2 developments and demonstrations, the requirements captured for WP 1 were derived from the High Level (HL) requirements generated upfront. Thus the requirements defined here-in provide a higher level of detail for the overall project goal related HL requirements as stated in the HL requirements document. The WP 1 Task 1.1 requirements cover the following areas:

- System Security
- System safety
- Real time capabilities
- Energy usage and computation efficiency
- System Power Supply efficiency and flexibility
- AMSPS functionality and safety
- System robustness and fault tolerance
- Performance predictability
- Variability adaptability
- System service criticality capabilities

3.2 Examples of selected requirements

Requirements TL-REQ-WP01-110 through TL-REQ-WP01-119 denote a requirements group dedicated to the extension developments based on the results of the GENESYS, INDEXYS and ACROSS projects. These developments comprise “System Safety”, “System Service Criticality Capabilities”, Real-Time Capabilities”, “System Robustness and Fault-Tolerance”, “Performance Predictability” and “Variability and Adaptability”. Connected to the appropriate HL requirements this requirements group will form the basis for the extension of the ACROSS MPSoC w.r.t. to increasing the performance by replacing the ALTERA NIOS cores by more performant CPU cores.

On the other hand, the requirements listed as TL-REQ-WP01-106 to TL-REQ-WP01-109 are related with new extensions of highly accurate and reliable time transfer mechanisms. They open the door to distributed safety critical systems providing time and frequency references distributions. These requirements focus on the issues needed to perform safe and dependable communication of time information using FPGA hardware and embedded firmware. Monitoring information mechanism is also included in order to provide inputs of synchronization status and alarms in case of malfunctions.

To make functionalities in systems available to other systems it is important that there is a well-defined framework of how the communication should be handled. This includes interoperability, integrability (in existing and new systems), service discovery and composition of services, there should also be no limitation in scalability of system size. These aspects are covered by requirements TL-REQWP01-001 to TL-REQ-WP01-006 and other supporting requirements.

4. Discussion

4.1 Requirements concerning time triggered data communication

The requirements comprise several items related to the results of the ACROSS architecture. This approach consisted of an ALTERA FPGA based development using ALTERA eight NIOS cores as CPUs. The goal of the project was first of all to demonstrate that such development would be possible and definitely could argue to deliver design advantages in cross domain industrial use cases. It was not a target of the project to provide a highly performant solution. Certainly it cannot be expected that the overall system would provide a performance beyond the capabilities of the NIOS cores. Another goal in ACROSS was to demonstrate that aerospace grade certification for this multicore design would be possible.

Both targets were achieved with excellent results. What remained was to connect high performant CPUs for different purposes to the network on chip (i.e. digital signal proceeding CPU or CPU for advanced video processing etc.). This would significantly increase the performance of such design and make the result significantly more attractive for various applications i.e. in the aerospace industrial domain. To enhance the results gained from ACROSS is one of the dedicated goals of EMC^2. Service oriented, modular data communication architectures for various applications in the safety relevant control domain provide significant added value due to their flexibility, their cross domain usability and due to increasing reliability when re-using proven building blocks to generate new applications.

In addition the EMC^2 WP 1 T1.1 targets to connect the ACROSS approach to other data communication technologies in order to allow as much as possible to re-use existing designs in any kind of application.

4.2 Requirements concerning interoperability, discovery and composition

Results from the Arrowhead project will be a valuable source of information and hopefully the Arrowhead framework can be applied (with modifications) in some application areas within the EMC^2 project.

4.3 Requirements concerning reliable time transfer distribution

The requirements comprise several issues focusing on the safe and reliable transmissions of time and frequency references. This approach opens the door to the development of distributed, safety critical system developments thanks to the utilization of a uniform and global time reference. This is possible because of a high synchronization accuracy and deterministic mechanism for the dissemination of these references.

4.4 Requirements concerning variability, adaptability

In order to enable systems for variability and adaptability one main requirement is the automatic reconfiguration of software/hardware systems (at local and/or distributed and global levels). This requirement connects to the EU EPiCS project, which focuses on concepts and foundations for self-aware and self-expressive systems with novel hardware/software platform technologies and architectures for engineering autonomic compute nodes and networks. Hence, results of the EPiCS project are incorporated into specific system requirements.

One main goal is to provide a middleware [43] for sensor networks [44] which carries out system reconfigurations independent from the executed application. In this way, the system is adapted to dynamic changes (e.g., various data transfers from sensors) and the application is capable to perform its tasks. In particular, computer vision applications (but also applications based on other sensors) benefit from this system behavior. Suitable hardware units (e.g., SoCs with multicore CPUs with GPU) are essential to provide sufficient computational resources for applications.

The automatic reconfiguration shall enable the system for the efficient **usage** of available hardware units, data exchange between different hardware units, and handover of tasks from one hardware unit to the next as well as graceful degradation in overload situations. These system properties foster complex sensor networks in different domains. Possible frameworks and middle-wares will have to consider run-time system properties such as the resource usage (e.g., CPU utilization, memory capacity and memory/ network bandwidths), delivered QoS (e.g., application dependent quality metrics like confidence levels) and other metrics and relate these to the desired application in way that enables to face these challenges more easily from application designer’s perspective.

5. Conclusions

A short overview has been given on the state of the art in embedded system architectures. The trend in this field is a move towards Service Oriented Architectures because they allow for greater flexibility and modularity, which in turn can result in better maintainability. Several European projects have made progress towards implementing Service Oriented Architectures for networked embedded systems, e.g. industrial control systems.

The requirements captured within this work package and task aim at enhancing SoA based developments and building blocks already available today towards higher overall performance, insertion into more new industrial domains, including numerous safety relevant controls, closing potential gaps in the approach such as providing modelling and tools to make better use of this promising approach. Enhancing the portfolio of services will in the long run provide designers a larger repository of pre-evaluated and proven building blocks reducing the cost of development, the integration efforts, and increasing the flexibility. Developments using a SoA can more easily be enhanced and can integrate new functionality at lower cost compared to resign from using such opportunity. In addition numerous developments can more easily be used as well in other industrial domains due to the common approach when basing developments on a service oriented design.

With the results from related projects, we will in this work package work towards one (or several) SoA-based system solutions for environments where multicore architectures are used in mixed critical applications.

6. References

[1] T. Erl, SOA Design patterns, Prentice Hall PTR, 2009.

[2] B. Mayer, Eiffel: The Language, NJ, USA: Prentice-Hall, Inc., 1992.

[3] C. A. R. Hoare, "An Axiomatic Basis for Computer Programming," *Commun. ACM,* vol. 12, pp. 576--580, 1969.

[4] *ADA 2012, ISO/IEC 8652:2012, standard.*

[5] J. Barnes, High Integrity Software: The SPARK Approach to Safety and Security, Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 2003.

[6] P. Couq, F. Kirchner, N. Kosmatov, V. Prevosto, J. Signoles and B. Yakobowski, "Proceedings of the 10th International Conference on Software Engineering and Formal Methods," in *Frama-C: A Software Analysis Perspective*, Thessaloniki, Greece, 2012.

[7] "http://www.omg.org/spec/OCL/," 2014. [Online]. Available: http://www.omg.org/spec/OCL/.

[8] G. Behrmann, A. David, K. G. Larsen, O. Möller, P. Pettersson and W. Yi, "Proc. of 40th IEEE Conference on Decision and Control," in *Uppaal - Present and Future,* 2001.

[9] B. Bernhard, R. Hähnle and P. H. Schmitt, Verification of Object-oriented Software: The KeY Approach, Berlin, Heidelberg: Springer-Verlag, 2007.

[10] World Wide Web Consortium, "SOAP Version 1.2 Part 1: Messaging Framework (Second Edition)," 27 4 2007. [Online]. Available: http://www.w3.org/TR/soap12-part1/. [Accessed 24 5 2014].

[11] World Wide Web Consortium, "Web Services Description Language (WSDL) Version 2.0 Part 1: Core Language," 26 6 2007. [Online]. Available: http://www.w3.org/TR/wsdl20/. [Accessed 24 5 2014].

[12] OASIS, "UDDI Version 3.0.2," 19 10 2004. [Online]. Available: http://uddi.org/pubs/uddi-v3.0.2-20041019.htm. [Accessed 26 5 2014].

[13] Microsoft, "Windows Communication Foundation," [Online]. Available: http://msdn.microsoft.com/en-us/library/dd456779(v=vs.110).aspx. [Accessed 19 5 2014].

[14] OPC Foundation, [Online]. Available: https://opcfoundation.org.

[15] OPC Foundation, "OPC Foundation," [Online]. Available: https://opcfoundation.org/about/opc-technologies/opc-ua/. [Accessed 18 5 2014].

[16] M. Stopper and B. Katalinic, "Service-oriented architecture design aspects of OPC UA for industrial applications," in *Proceedings of the International MultiConference of Engineers and Computer Scientists, IMECS*, 2009.

[17] W. Mahnke, S.-H. Leitner and M. Damm, OPC Unified Architecture, Springer Publishing Company, Incorporated, 2009.

[18] OASIS, "Strategy," [Online]. Available: https://www.oasis-open.org. [Accessed 19 5 2014].

[19] T. Andrews, F. Curbera, H. Dholaika, Y. Goland, J. Klein, F. Leymann, K. Lui, D. Roller, D. Smith, I. Trickovic and S. Weerawarana, *Business Process Execution Language for Web Services,* S. Thatte, Ed., 2003.

[20] D. Jordan, J. Evdemon, A. Alves, A. Arkin, S. Askary, C. Barreto, F. Curbera, M. Ford, Y. Goland, A. Guízar, N. Kartha, R. Khalaf, D. König, M. Marin, S. Thatte, D. Van der Rijn, P. Yendluri and A. Yiu, *Web services business process execution language version 2.0,* vol. 11, OASIS standard, 2007.

[21] OASIS, "Devices Profile for Web Services Version 1.1," 1 7 2009. [Online]. Available: http://docs.oasis-open.org/ws-dd/dpws/1.1/os/wsdd-dpws-1.1-spec-os.pdf. [Accessed 26 5 2014].

[22] H. Bohn, A. Bobek and F. Golatowski, "SIRENA-Service Infrastructure for Real-time Embedded Networked Devices: A service oriented framework for different domains," in *Networking, International Conference on Systems and International Conference on Mobile Communications and Learning Technologies*, 2006.

[23] F. Jammes, A. Mensch and H. Smit, "Service-oriented device communications using the devices profile for web services," in *Proceedings of the 3rd international workshop on Middleware for*

pervasive and ad-hoc computing, 2005.

[24] ITEA, "SIRENA," [Online]. Available: https://itea3.org/project/sirena.html. [Accessed 27 5 2014].

[25] R. Kyusakov, J. Eliasson, J. Delsing, J. van Deventer and J. Gustafsson, "Integration of wireless sensor and actuator nodes with IT infrastructure using service-oriented architecture," *Industrial Informatics, IEEE Transactions on,* vol. 9, no. 1, pp. 43-51, 2013.

[26] Z. Shelby, Sensinode, K. Hartke, C. Bormann and U. B. TZI, "Constrained Application Protocol (CoAP) draft-ietf-core-coap-18," 2013. [Online]. Available: https://datatracker.ietf.org/doc/draft-ietf-core-coap/. [Accessed 17 5 2014].

[27] Z. Shelby Sensinode, "RFC-6690," [Online]. Available: http://www.rfc-base.org/rfc-6690.html.

[28] F. J. Cazorla, E. Quiñones, T. Vardanega, L. Cucu, B. Triquet, G. Bernat, E. Berger, J. Abella, F. Wartel, M. Houston, L. Santinelli, L. Kosmidis, C. Lo and D. Maxim, "PROARTIS: Probabilistic Analyzable Real-Time Systems," *AMT Transaction on Embedded Computer Systems - Special section on Probabilistic Embedded Computing,* 2013.

[29] R. Davis, T. Vardanega, J. Andersson, F. Vatrinet, M. Pearce, I. Broster, M. Azkarate-Askasua, F. Wartel, L. Cucu-Grosjean, M. Patte, G. Farrall and F. Cazorla, "PROXIMA: A Probabilistic Approach to the Timing Behaviour of Mixed-Criticality Systems," *Ada User Journal,* vol. 2, pp. 118-122, 2014.

[30] A. Burns and R. Davis I., "Mixed Criticality Systems - A Review," Department of Computer Science, University of York, UK, York, 2014.

[31] R. M. Pathan, "Fault-Tolerant and Real-Time scheduling for Mixed-Criticality Systems," *Real-Time Systems: The International Journal of Time-Critical Computing,* vol. 50, no. 4, pp. 509-547.

[32] "XACML: eXtensible Access Control Markup Language, The OASIS technical committee," 2005.

[33] E. Rescorla and N. Modadugo, *Datagram Transport Layer Security Version,* Internet Engineering Task Force, 2012.

[34] J. Schneider and e. al., *Efficient XML Interchange (EXI) Format*, 2014: W3C Recomendation .

[35] D. Hardt, "The oAuth 2.0 Authorization Framework," in *Internet Engineering Task force RFC 6749,* 2012.

[36] L. Seitz, G. Selander and C. Gehrmann, "Authorization framework for the Internet-of-Things," in *World of Wireless, Mobile and Multimedia Networks,* 2013.

[37] Z. He and T. Voigt, "Droplet: A New Denial-of-Service Attack on Low Power Wireless Sensor Networks," in *10th International Conference on Mobile Ad-hoc and Sensor Systems*, 2013.

[38] L. Wallgren, S. Raza and T. Voigt, "Routing Attacks and Countermeasures in the RPL-Based Internet of Things," *International Journal of Distributed Sensor Networks,* 2013.

[39] A. Cannata, M. Gerosa and M. Taisch, "SOCRADES: A framework for developing intelligent systems in manufacturing," in *IEEE International Conference on Industrial Engineering and Engineering Management (IEEM 2008)*, 2008.

[40] A. Pedro, D. Pereira, L.M. Pinho and J.S. Pinto, "A Compositional Monitoring Framework for Hard-Real-Time Systems," in NASA Formal Methods Symposium 2014 (NFM14), Springer International Publishing, 29 of April 2014, LNCS 8430, pp 16-30, Huston, Texas, USA.

[41] A. Pedro, D. Pereira, L.M. Pinho and J.S. Pinto, "Towards a Runtime Verification Framework for the Ada Programming Language," in Reliable Software Technologies - Ada-Europe 2014, Springer International Publishing, 16 of March 2014, LNCS 8454, pp 58-73, Paris, France.

[42] W. Wu, D. Kumar, B. Bonakdarpour, and S. Fischmeister, "Reducing Monitor Overhead by Integrating Event- and Time-triggered Techniques," in Proceeding of the International Conference on Runtime Verification 2013.

[43] M. Garcia Valls, I. Lopez and L. Villar, "iLAND: An Enhanced Middleware for Real-Time Reconfiguration of Service Oriented Distributed Real-Time Systems," Industrial Informatics, IEEE Transactions on , vol.9, no.1, pp. 228-236, 02 2013.

[44] B. Dieber, L. Esterle and B. Rinner, "Distributed resource-aware task assignment for complex monitoring scenarios in Visual Sensor Networks," in Sixth International Conference on Distributed

Smart Cameras (ICDSC), 2012.

7. Appendix

7.1 Abbreviations

Table 1: Abbreviations

Abbreviation	Meaning
EMC^2	Embedded multi-core systems for mixed criticality applications in dynamic and changeable real-time environments
µC	Micro-Controller
SOA	Service Oriented Architecture
CoAP	Constrained Application Protocol
DPWS	Device Profile for Web Services
XML	Extensible Markup Language
XSD	XML Schema Definition
WSDL	Web Services Description Language
SOAP	Simple Object Access protocol
OLE	Object Linking and Embedding
OPC	OLE for process control
OPC UA	OPC Unified Architecture

7.2 Technical requirements derived from EMC^2 high level requirements

Technical requirement list in excel format.