Internet of Systems Engineering: Simplifying the Development of Complex Engineering Systems

Michael Wagner * — Ina Schieferdecker * — Tom Ritter * — Christian Hein

**Fraunhofer FOKUS*
Kaiserin-Augusta-Allee 31
10589 Berlin, Germany

Michael.Wagner@fokus.fraunhofer.de
Ina.Schieferdecker@fokus.fraunhofer.de
Tom.Ritter@fokus.fraunhofer.de
Christian.Hein@fokus.fraunhofer.de

ABSTRACT: *A particular area of strength for the European industry is the development of value-added complex products such as aircrafts and automobiles. Challenges in coordination and information exchange as well as full verification and validation arise besides distribution. The internet of things gives the opportunity for solving the challenges by connecting designer, physical devices and models on the same level to the internet of systems engineering allowing exchangeability. In combination with distributed simulation and testing over the internet technological advantage is provided to the European systems integrator.*

This paper presents recent results of the European FP7 project SPRINT : Software Platform for Integration of Engineering and Things (SPRINT 2013). Grant Agreement No 257909

KEY WORDS: *Systems Engineering, Collaborative Development, Semantic Mediation, Linked Data, Integrated framework.*

1. Introduction

A particular area of strength and interest for the European industry is the development of value-added complex products such as aircrafts and automobiles characterized by the innovative integration of mechanical and electronic components. The increase in electronics and software content inside traditional mechanical products increases their complexity further and faces substantial technological and process-oriented changes in their development. The ever increasing complexity of the products has a strong impact on time to market, cost and quality, the main drivers of world-wide competitiveness.

Indeed, the economic aspects of hardware/software subsystems - called *Embedded Systems* - running within these products are remarkable. For example, avionics costs are about 30% of the overall cost of an aircraft and embedded systems represent about 40% of avionics cost. Moreover, the importance of embedded software has been rising almost exponentially over time. The size of software doubles with each new Airbus program: from the 1980s Airbus A300 which had about 200kB of software on board, to the A380 whose software size is in the range of millions of lines. For this aircraft, a single line of certified level A software code is estimated to cost about 100€ thus yielding an overall cost for software of hundreds of millions of Euros.

Further, the design and supply chains of these products have become increasingly complex with suppliers taking a greater part in the design process of the overall system, often taking full responsibility for the design and manufacturing of major subsystems. Airbus uses about 3000 companies as suppliers in its supply chain. This diversity of companies and countries involved in the development of an aircraft brings with it an almost unmanageable variety of tools that are aimed at specialized tasks and confined to single countries and/or companies.

2. Problem

There are two major roadblocks to achieve the degree of industrial efficiency needed to develop the next generation of complex software-intense value-added European products:

– Controlling the design process in present environments requires great *coordination and information exchange* that is almost impossible today. This situation is compounded by the geographical distribution of design teams of the main system developer companies and the outsourcing trend that requires managing the relationship with distant design teams that need non-trivial support from the primary company.

– *Full verification and validation (V&V)*: Due to the tight dependency in physical location, the integration of these products is carried out today at the last phase of the engineering lifecycle - the integration phase - when all subsystems are available and the company who is responsible for the final delivery of the product assembles it in the almost final configuration. Any error discovered at this late stage is likely to cause major delays and cost overrun.

Presently the prevalent approach is a "siloed" one where each stage of the design is mostly confined to a single phase and to a single company (see Figure 1). In this approach, after a short common system design stage in which the functional system architecture is defined and the work on different components is allocated among the many different teams, the work is performed by geographically distributed teams for a time period that may span many months that may even be in different time zones. Each team is working on a given component with their own development equipment, laboratories, hardware and software design methods, and tool providers. The appropriate testing and verification procedures are almost always delayed to the late point in time when the system is completely assembled. Already today - see e.g. the software integration issues the A380 was facing - the delayed verification process results many times in a large amount of redesign thus causing serious slips in delivery schedules and consequent penalties. Clearly such an approach is untenable for next generation designs.

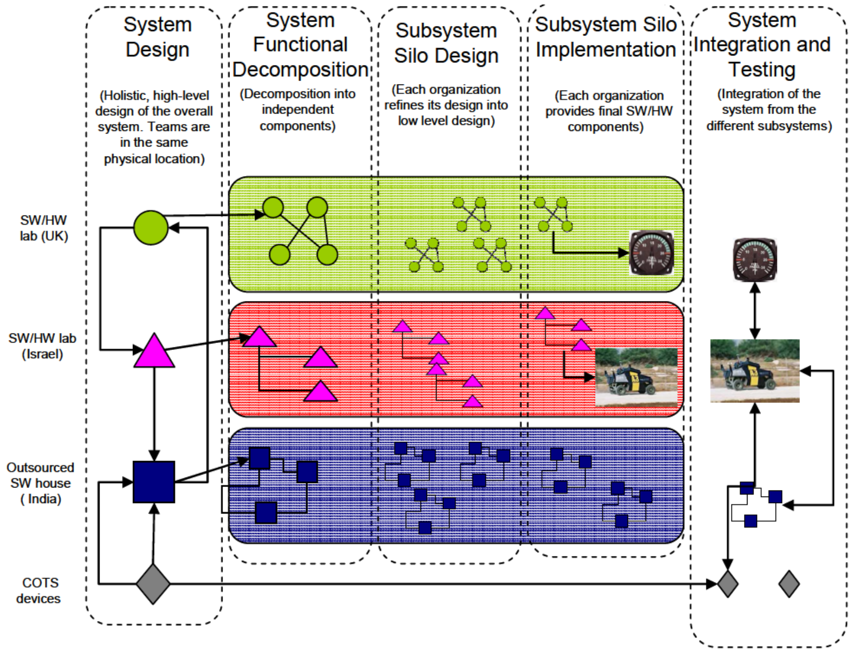

Figure 1 "Siloed" approach before SPRINT (Shani et al. 2014)

As a study performed by the business research firm Venture Development Cooperation had shown (VDC 2007), market desperately needs new collaboration, lifecycle management tools and verification methodologies for complex products made of a number of collaborating devices and developed by many different and remote teams. Industries such as telco service providers, retail and financial services providers exploit already the Internet in new and creative ways that dramatically reduce costs while increasing revenues. However, the system engineering industry is still constrained by the barriers described above created by the physical location of its engineers, their development and quality assurance processes, their design and development environments, and the physical testing and simulation labs.

The design, development, and operation of systems engineering services via the Internet impose challenges in seamless and timely information exchange and work coordination, in device integration and virtualization, and in early and continuous V&V. These challenges open enormous opportunities for new methods and tools in the design and implementation of complex products that leverage the concept of Internet of Things, building upon the already established concept of Internet of Services, as an infrastructure that allows rich information exchange among widely distributed devices and design teams as schematically depicted in Figure 2.

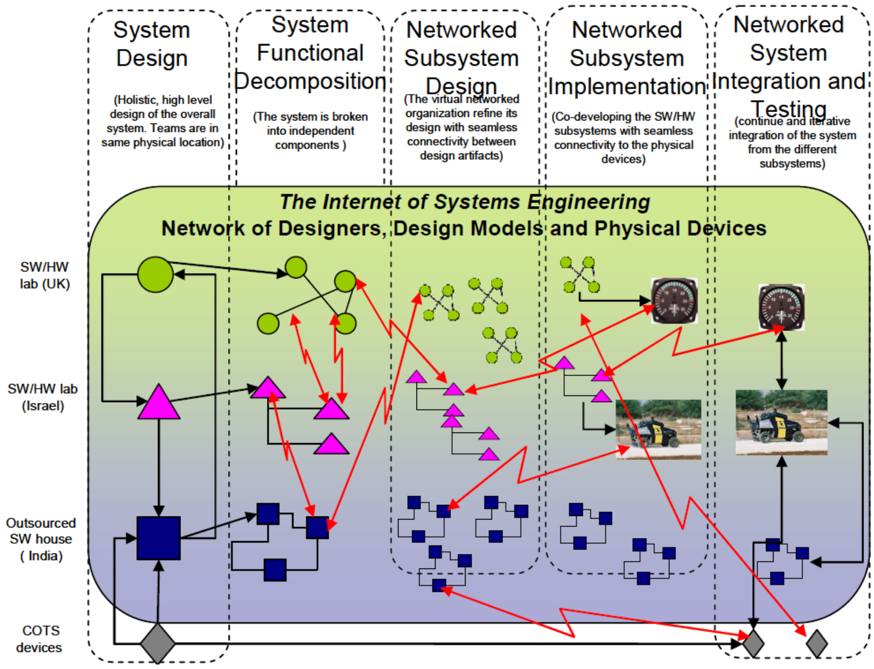

Figure 2 **Applying the Internet of Things and the Internet of Services to Systems Engineering (Shani *et al.* 2014)**

This infrastructure should also include novel approaches for early and continuous V&V that involve virtual integration of mathematical models of components under design and of physical components already manufactured to alleviate the problem of late error discovery. Because of the geographical distribution of design teams, this integration must leverage the Internet in a novel and unconventional way whereby artefacts interface to the network and interact with virtual components applying the concept of Internet of Things in a radically innovative way paving the way for a sensible shift in the paradigm in which complex products are designed and tested.

3. Solution

The approach of the SPRINT project was to create an *Internet of Systems Engineering* based on an Internet of Services-infrastructure, intended to provide a simple and straightforward blending of all "things" pertaining to a complex system such as: sensors, actuators, electro-mechanical components, processing elements, system design tools and their models and development teams into a common platform with highly flexible collaboration and interoperability capabilities to support geographically-distributed teams through the entire cycle of complex system development: design, development, integration, V&V, and deployment.

To achieve this, the Internet of Systems Engineering was build built upon three main pillars:

1. *Internet of Physical Devices*, where physical components are connected via the Internet to facilitate distributed design, remote simulation and testing; this is a concept that requires the development of new methods to model and evaluate correctness of complex interactions among physical devices.

2. *Internet of Design Model Elements*, where model elements, regardless of their origin (tool, location, organization), can be located, referenced, shared and used across the Internet. For early verification, we need also methods to interface Physical and Virtual Devices.

3. *Internet of Designers*, where engineers no matter where they are located can collaborate and cooperate over the Internet in an efficient and productive manner as if they were on the same organization and physical location.

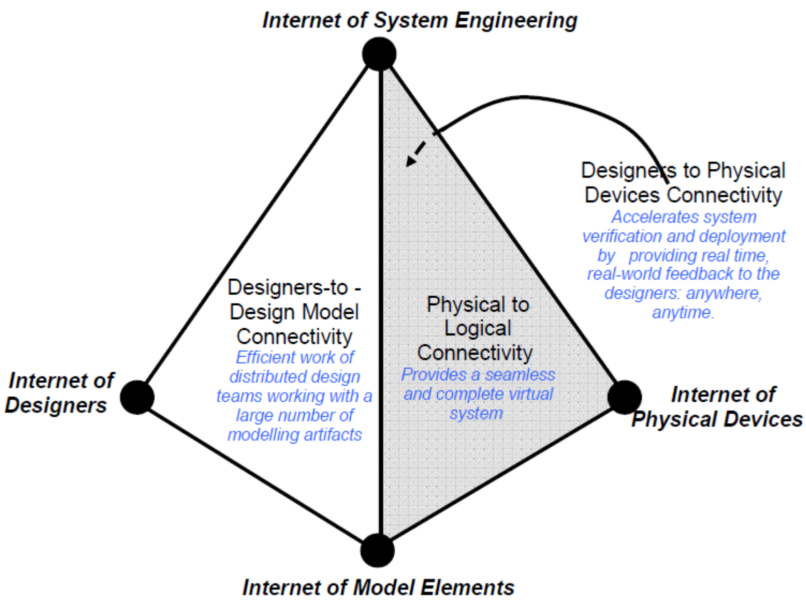

Figure 3 The tree pillars of the Internet of Systems Engineering (Shani et al. 2014)

These three pillars are logically connected as shown schematically in Figure 3. They are represented in Figure 3 as vertices of a pyramid connected by edges that signify their interrelationship and the capabilities offered to designers. By threating them all as equal they can be connected via the Internet of Things to create a solution to overcome the two major roadblocks for efficient development of next generation value-added complex European products. To achieve this, the Sprint project adopted the IoT – Architecture (Carrez *et al.* 2013) proposed by the IoT-A project enabling different domains to extend a common architecture and understanding for the internet of things, this way cross benefiting from each other.

4. Underlying Magic

SPRINT achieved its success through providing three mayor capabilities: *Collaboration on and Distribution of Information*, *Semantic Mediation* and *Distributed Simulation and Testing* for the development of interconnected embedded systems.

4.1. *Collaboration on and Distributed of Information*

Along the adoption of the Open Services for Lifecycle Collaboration initiative (OSLC), which has substantially progressed through the support of SPRINT and is already under standardization by OASIS, the Internet of Systems Engineering is using *semantic Web* and *linked data* technologies. OSLC has been strongly supported by IBM in its open Jazz platform. IBM has implemented the OSLC protocol over all of its product lifecycle management (PLM) tools. OSLC gets more and more adapted as it is now also implemented by the ModelBus integration framework.

OSLC is an example of the value of linked data as defined by W3C (see Figure 4). Design resources such as requirements, specifications, project management artefacts, models and their elements, or documents are all internet artefacts, having an accessible URL. They can link each other and be linked by any application or by any Web page in the internet.

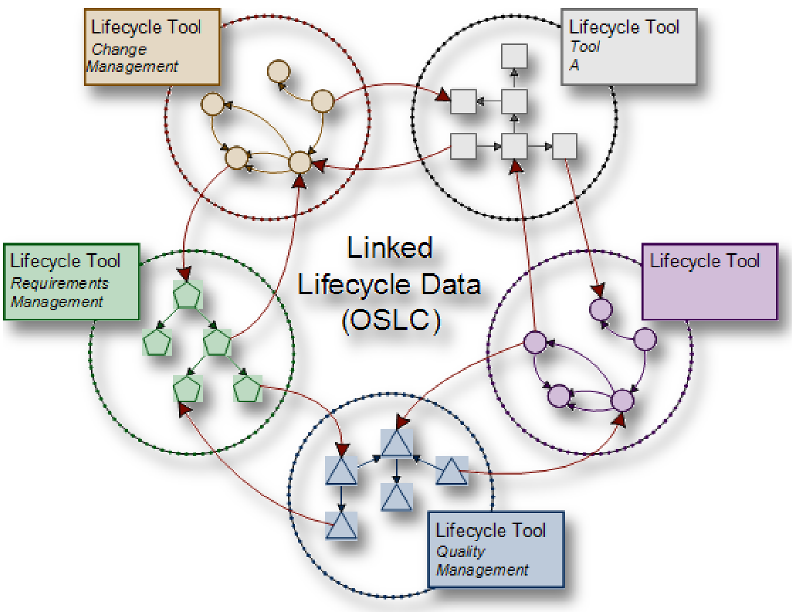

Figure 4 Use of Linked Data in OSLC (Shani *et al.* 2014)

In this way, tools "open up" their content. The standard Resource Description Framework (RDF) of linked data provides communicable representations of the information and the access to the information via standard communication protocols. Any tool can thus "look into" and make use of the information such as model elements in other tools. Furthermore, some tools may serve as an "index" of the design resources accessible via the URLs. By that, a holistic and current view of the ever increasing ecosystem of models and PLM artefacts can be provided.

4.2. ***Semantic Mediation***

As there is only some level of semantic information in OSLC (such as in the architecture management or quality management categories), the Internet of Systems Engineering uses in addition *semantic mediation techniques* to enable a common and deeper understanding on the things, artefacts, devices, software etc. being considered. In order to support this, SPRINT requires that each tool not only opens up its content, but also represents the content by concepts defined in an ontology. The ontologies can be defined by the Web Ontology Language (OWL), by an RDF schema or in other ontology specification language as proposed by W3C. These ontology specifications are too defined in RDF and can hence be shared like the other artefacts in the Internet of Systems Engineering.

By using ontologies, the content shared by a tool becomes meaningful and reusable in a tool-independent and non-proprietary way. If two distinct tools expose their content in RDF, and associate them with a semantic interpretation by an ontology, it is possible to mediate the content from one tool to the content in another tools. The rules of mediation can also be defined by ontology, so that interpretation and mediation of content is realized by the same set of semantic technologies.

This approach also allows identifying common representations for commonalities of ontologies (or of the content in tools). For example, hardware/software architecture tools can have a common ontology as simple and as general as the OSLC architecture management ontology, or a more specific one, such as the Systems Modeling Language (SysML) ontology.

Figure 5 Semantic Mediation utilizing BSO (Shani *et al.* 2014)

Along this, SPRINT defined the basic structure ontology (BSO) for the basic structure of models. BSO is considered sufficiently common to the DESYRE tool by ALES, the Wolfram SystemModeler by Wolfram, and the Rhapsody tool by IBM, see Figure 5.

In fact, BSO is not a single plain ontology, but is defined as a hierarchy of ontologies as depicted in Figure 6. This figure shows different levels of commonalities between ontologies, representing different ranges of interoperability between the tools. There is even some interoperability among tools that have little in common. Yet, even among tools with great level of commonality, the interoperability is not 100% since there are always some unique capabilities of tools that are not meaningful in other tools. So, sharing whatever is possible from one tool to another tool, makes it possible to easily apply the unique capabilities of the other tool to the model. Without this sharing, it would have been quite expensive to make it possible to use that unique capability (e.g., static analysis, or continuous event simulation, etc.).

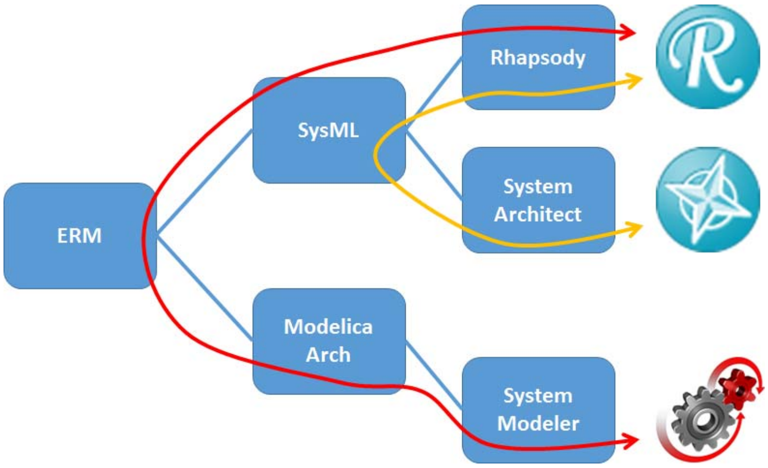

Figure 6 Semantic mediation by hierarchical ontologies (Shani *et al.* 2014)

4.3. *Distributed Simulation and Testing*

Last but not least, SPRINT enables early V&V for the integration of collaboratively designed system components by real-time distributed simulation. Such simulation capabilities provide a number of advantages for V&V like:

– Partners might not be willing to share models with each other for IP reasons. If a system of subsystems that are modelled by different partners having different tools are to be simulated, that simulation must be distributed. It can e.g. be done in a distributed manner with each subsystem simulation running at the respective partner's site.

– Physical devices cannot easily be shared. Due to physical sizes of components, it might be difficult to ship a device in a certain location to perform testing. Moreover, shipment would introduce extra delays in the design process. Last, the availability of physical devices might be limited. Distributed simulation, instead, would not only enable the integration while maintaining the device at the original site, but also make it available for other design activities.

– All partners might not have access to or the possibility to use all the tools used by other partners in the project due to licensing costs, lack of know-how, or alike. This case can also be solved by employing distributed simulation, but it can also be solved by using hosted simulation. For that to work all participating tools needs to be able to export to a common executable model exchange format and each partner needs to have access to one tool that can perform hosted simulation using that format.

SPRINT supports the Internet of Systems Engineering by services for *Distributed (Real-Time) Simulation with Simulated or with Physical Devices* and for *Integration Testing of Physical Devices over the Internet*.

In *distributed simulation* (see Figure 7), a user can simulate a system composed of several subsystems, where each of the subsystems might have been created in a different tool. Each subsystem is simulated in its own simulator that can reside at different geographical locations due the aforementioned reasons.

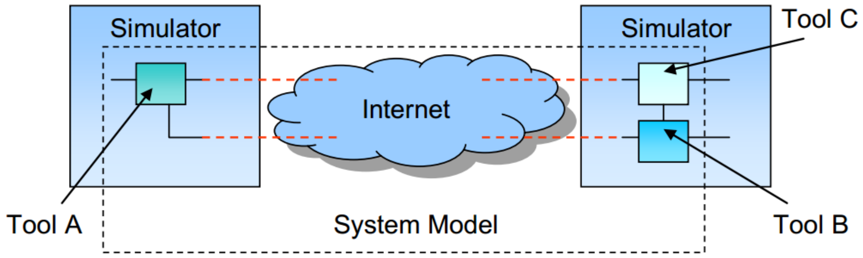

Figure 7 Distributed simulation with simulated devices (Tronarp *et al.* 2014)

In case of *distributed (real-time) simulation with physical devices* (see Figure 8), a user can simulate a system composed of several subsystems, where some of the subsystems exist as physical devices, while the others are simulated only. The physical devices might for example be sensors or real implementations of the components. The main reason for doing this is to perform early integration testing.

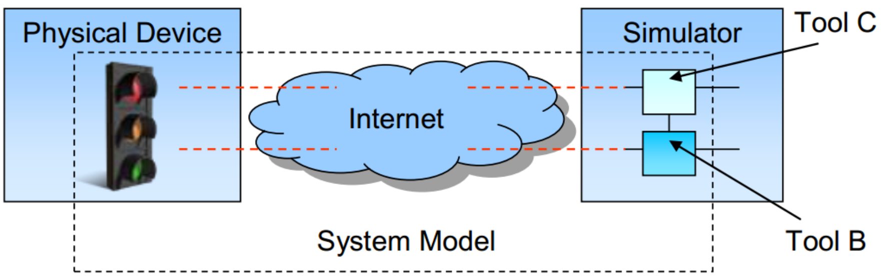

Figure 8 Distributed simulation with physical devices (Tronarp *et al.* 2014)

In *integration testing of physical devices over the Internet* (see Figure 9), all subsystems of the system may exist as physical devices and the user wants to do integration testing of the devices over the internet. All the physical devices reside at different locations and the complete system is connected over the Internet.

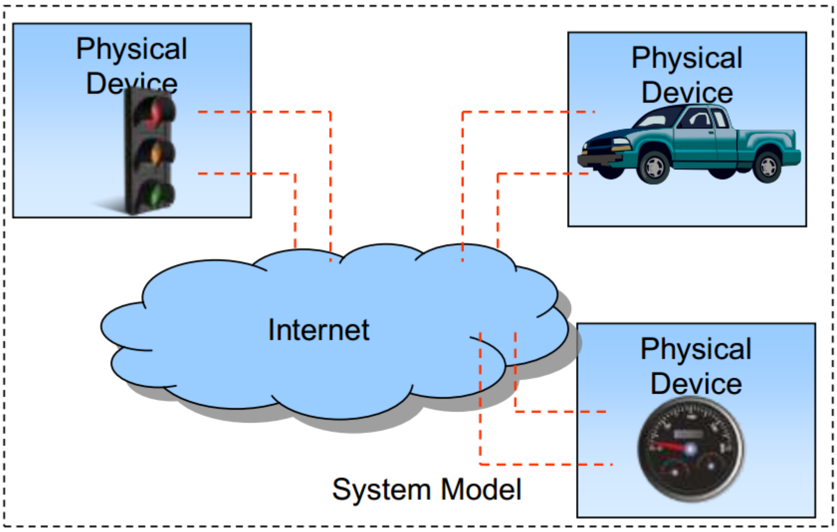

Figure 9 Integration Testing of Physical Devices over the Internet (Tronarp *et al.* 2014)

5. Current Status and Accomplishments

The Internet of Systems Engineering supports new approaches for the development of highly complex, software-intensive, networked embedded systems. It enables new engineering services along the interconnection of hardware/software components, subsystems, systems and physical devices and for the efficient collaboration of remote teams with different tool landscapes. The provisioning of distributed V&V services allows for an early and continuous quality assurance in systems integration by separated development teams.

The SPRINT project developed the underlying concepts for the Internet of Systems Engineering and demonstrated successfully their use for the development of complex innovative products. By wrapping physical devices with engineering services and work flows, SPRINT engineers were able to work effectively, coherently and efficiently although they operated in quite different teams across multiple countries and applying different tool sets.

6. Bibliography/References

Braun T., Diot C., Hoglander A., Roca V., An experimental user level of implementation of TCP, rapport de recherche no. 265. septembre 1995, INRIA.

Carrez F. et al., IoT-A Deliverable D1.5 Final architectural reference model for the IoT v3.0, file:///C:/Users/mwa/Downloads/D1.5.pdf, 15 july 2013

OASIS, Organization for the Advancement of Structured Information Standards, https://www.oasis-open.org/, 2014

OWL, Web Ontology Language, http://www.w3.org/2001/sw/wiki/OWL, 27 octobre 2009

RDF, Resource Description Framework, http://www.w3.org/RDF/, 10 february 2004

Tronarp O., Boccabella S., D’Angelo M., Ferrari A., Mignogna M., Sofronis C., SPRINT D4.17 Real time simulation service definition, 7 february 2014.

Shani U., Raiend K., Wadler D., Hedberg D., Wagner M., D'Angelo M., Tronarp O., SPRINT D5.11 Final Report, 17 march 2014.

SPRINT, EU FP7 Project on a Software Platform for Integration of Engineering and Things, http://www.sprint-iot.eu, 2010-2013.

SysML, Systems Modeling Language, http://www.omgsysml.org/, 8 june 2012

VDC, Venture Development Corporation Embedded Hardware and Systems Practice. Report, http://www.vdccorp.com/_documents/pressrelease/press-attachment-1333.pdf, 2007.

OSLC, Open Services for Lifecycle Collaboration initiative, http://open-services.net, 2013.

W3C, World Wide Web Consortium, http://www.w3.org/, 2014.