Feasibility Study in the use of contract-based approaches to deal with safety-related properties in CPS

Daniela Cancila
CEA, LIST, CEA Saclay - F91191 Gif-sur-Yvette Cedex; email: daniela.cancila@cea.fr
Elie Soubiran
Technological Research Institute SystemX - Alstom
Transport; email: elie.soubiran@{irt-systemx.fr,transport.alstom.com}
Roberto Passerone
Dipartimanto di Ingegneria e Scienza dell'Informazione - University of Trento,
Italy; email: roberto.passerone@unitn.it

Abstract

This work concerns a feasibility study on the use of contract-based approaches as a means of reasoning and understanding a cyber-physical system (CPS) which should meet safety properties. We show the problems, the analysis methodology and the results on a railway industrial system case study. Our results suggest that contract-based design provides a rigorous approach for reasoning at the interaction of safety-related properties in CPS.

Keywords: contract-based approach, CPS, Railway system, mixed-critical and safety-related properties.

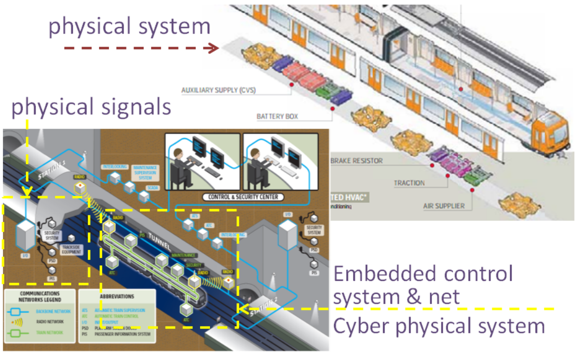

Figure 1: Image extracted from 'Metropolis And Metro Train Solution' by Alstom [11]

1 Introduction

In the last decade, Cyber-Physical Systems (CPS) have assumed an increasingly significant role in a number of disciplines, especially in Computer Science, and form one of the cornerstones of the study of dynamical and heterogeneous systems. CPS combine signals from physical components with (embedded) software components and integrated circuits.

Historically, the term 'cyber-physical systems' was first introduced by H. Gill to broadly capture a similar meaning of the term 'cyberspace' and 'cybernetics' [1]. Since then, the term CPS has been widely adopted by the scientific community and, today, it appears as one of the main topics of the European projects (e.g., H2020, EIT ICT Labs).

Contract-based approaches are considered as a promising means to deal with CPS [2,3,4,5,6,7,8]. A contract is a pair (assumption, guarantee), where the guarantee specifies the functionality provided by a component to the environment; and the assumption sets forth the conditions required from the environment in order for the component to accomplish its guarantee [5]. The contracts, which are specifications on both physical and computational components, help us identify precisely the conditions for a correct interaction.

This position paper arises from the FSF project (*Fiabilité et Surêté de Fonctionnement* Reliability and Safety) [9]. The bulk of the FSF project deals with safety-related properties of a railway system that involves components, which have an inherent different nature and, to complicate the scenario further, combine different safety integrity levels (SIL) [10]. This work is a feasibility and preliminary study that explores a contract-based approach to deal with a seamless guarantee of safety-related properties from CPS design to execution platform. We feel that this approach can provide a simple, but firm, foundation to a rigorous approach for reasoning about the interaction of safety-related properties in CPS.

2 Case Study

Figure 1 shows both the mechanical part and the cybernetic part (i.e., command, control and supervision) of a railway system. A first command and control loop takes place within train units, where embedded software subsystems ensure automatic train driving and protection. These subsystems are mostly safety critical and shall furthermore consider real-time constraints. A second loop takes place at the line level, and is concerned with line supervision (train-traffic, timetable, etc.) and focuses on operational performance.

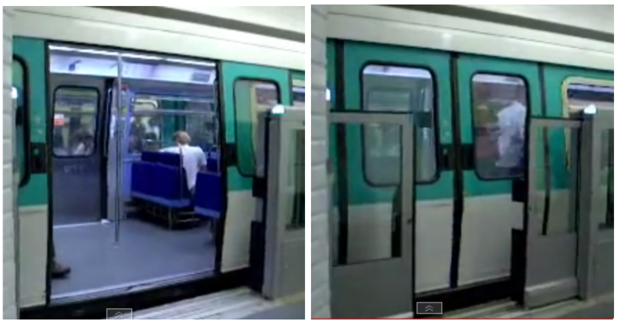

Figure 2: On the right, automatic opened doors, on the left, the platform doors are automatically closing (images extracted from youtube)

The case study considered in this paper is in the scope of the Communication Based Train Control (CBTC) system [12], and considers more precisely a subset of the Automatic Train Control subsystem (ATC). The associated operational scenario is the following: a train stops at a station that is equipped with a physical barrier and automatic doors, whose purpose is to protect passengers from the moving train (see Figure 2). In order to be able to operate train and platform doors, the doors of the train and the doors of the platform need to be aligned. At that point, both of them are automatically opened - thus allowing the passengers to get on and off the train. We will refer to this phase with the technical term *passenger exchange* in the rest of the paper. Finally, the train is authorized to move on if and only if both platform and train doors are closed.

The function *passenger exchange* is an important functionality of the CBTC, and this case study is obviously representative of CPS. Indeed, it integrates not only computational and physical processes with feedback loops, but also the human factor. This function takes control of platform and train doors when the train is safely docked at a station; then it organizes the exchange of passengers (e.g. manage train and station doors opening/closing and doors blocking by passengers) while protecting them from any untimely train movement or non-aligned doors opening. It finally gives the departure authorization when all safety conditions are met.

In this CPS we find different levels that co-exist, each of them with its own needs, requirements, guarantees. For example (list non-exhaustive):

- the door presence sensor, which ensures that no passenger is blocked between doors;
- acoustic and visual signalization, placed both on the platform and train side, which warn about the closing and opening doors.

The operational phase linked to this case study is critical since doors are open and passenger can move freely between the train and the station. Thus, it is relevant to focus the study on safety related properties that may be expressed and refined through contract-based analysis. To do so, we propose to start from identified hazards that cause accidents and/or near-miss accidents, then to establish contracts between the system components to define the necessary conditions that ensure safety, and then to refine those contracts down to software components and their associated computation unit. Beyond characterizing functional behaviours that would ensure safety invariant, the goal of contracts here will also be in a near future to support non-functional properties refinement and analysis with for instance SIL allocation, failure rate and so on.

3 Methodology

The CPS is initially modeled in SysML in the Papyrus tool - thus providing a holistic view of the whole system. For the sake of industrial adherence and industrial transfer of our work, we exploit the Alstom methodology to develop the model [13, 14]. The next paragraph reports the main principles of the quoted methodology, freely extracted from the Alstom documents [13].

In the last years, Alstom has developed the *Advanced System Architect Program* methodology, known as ASAP methodology, to increase quality of the system specification. In the methodology, textual requirements are initially deployed on model elements and are then further specified and refined. The modelling approach is threefold:

- operational vision, which deals with objectives and missions (why);
- functional vision, which concerns the strategy to perform missions (what);
- constructional vision, which addresses elements required to perform functions (how).

Alstom adopts the standard SysML language to implement the ASAP methodology. This latter has been tested on the Rolling Stock railway system, from Customer requirements/needs to product solution [13]. Some interesting industrial feedback on the use of SysML is provided by M. Ferrogalini and J. Le Bastard [14].

As firstly introduced, the ASAP methodology allows us to deal with physical signals, business needs, system specification and requirements. Therefore, we strategically adopt the ASAP methodology to specify the SysML model at an early stage of the development phase of our use case. When we refine the model further, however, we should be able to capture some details and then a component-based system engineering (CBSE) methodology seems to fit this scope better. In that context, a functional architecture is designed within the functional viewpoint, then resulting functions are allocated to components which belong to the constructional viewpoint. Following the SysML language primitives, components are represented by blocks, data by types and data transmission by port and connectors.

Our work strengthen the ASAP and the CBSE methodologies with a contract-based design approach.

3.1 Contract Specification

We adopt a textual format to introduce contracts at the CPS level. This approach fits better with high-level requirements, which are usually expressed in natural language. Our notion of contracts is based on previous work [5, 6, 7]. To the best of our knowledge, the ASSERT FP6 European project was the first to structurally establish the deployment of contracts on UML ports (and its profiles such as SysML or MARTE) [5, 15]. After that, several European research projects have widely adopted the relationship contracts - UML (and profiles) ports and successfully converged on it (see, for instance, the CHESS Artemis project [16]).

An intriguing use of contracts as a means to establish a firm relationship between software and control in CPS design has been recently introduced in the literature by Derler et al. [7]. There, functionality and timing are correlated in each of four types of contract to design effective control loops. This approach leads precision as well as abstraction - thus being easily applied to our use case.

Moreover, contracts are on one hand a means to prove correctness of heterogeneous components (through the notion of composability [17]), and, on the other hand, to prove the faithful refinement between two abstraction levels of a design [6]. In order to ensure continuous and automatic verification throughout the specification, the design and implementation phases, we are forced to eventually specify contracts by a formal, and non ambiguous, language. At this step of the development, we envisage adopting a similar language to that introduced in the literature [18] and, more recently, adopted by the Autosar consortium [19].

When we refine the system further, we follow the Platform-Based Design approach (PBD) [20, 21, 22]. This approach has been widely adopted by the scientific and industrial community, albeit not without difficulties and following several approaches [23]. Nonetheless, PBD allows us to introduce a common semantic domain between different abstraction levels as well as different views of a design, which help to maintain a consistent view of the system.

3.2 HMI and contract visualization

From a visualization point of view, 2D or 3D representations could help the designers have a better grasp of their systems. More in particular, a 3D representation could help us (and final costumers) reason about the physical aspects of CPS. It would provide a mean to simulate the CPS regarding different operational scenario and their respective impact on contracts. However, when we deal with automatic verification, we consider SysML UML supporting 2D tools, such as Papyrus, Obeo Designer, IBM or Atego, which are easily customizable.

3.3 Safety and Certification

Safety issues have a prominent role, especially in those CPS which ought to entail a certification process. This is exactly the case of some functionality and mechanical components of our use case. For example the *Passenger exchange* functionality and the mechanical signalling components involve the highest safety integrity level.

Each company has its own *savoir-faire* to identify and analyze the safety properties. Usually, Safety engineer teams identify and deeply study accident scenarios and identify barriers that mitigate the risk to an acceptable level. For instance, in the case study, an accident could result from a train that departs when the door are not yet properly closed. A functional barrier is then identified and provides a safe departure authorization to the train.

The performed analysis should be compliant to the related safety norms and validated by an independent certification entity. In many cases, the results of that analysis take the form of requirements, which identify safety barriers, such as preventive and palliative ones (non-exhaustive list).

Safety requirements should be adequately taken into consideration in all development phases of the system: from the specification to maintenance. As a result, their traceability is a key component of methodologies oriented towards the development of critical systems.

4 Application to the Case Study

In many cases, current industrial processes provide a list of requirements in a textual format. Not only are these latter exploited/improved during all development phases, but they are also used during the certification/qualification phase: the validator checks that (textual) code is compliant with all (textual) requirements.

The companies, which base industrial systems specification and analysis on component-based approaches, often adopt a bidirectional tool from textual requirements space to design modeling space. Then, they deploy requirements to model elements.

Like the industrial practice, in our approach a requirement is initially imported by a textual document.

[Req.] *The Passenger exchange train control function shall determine which train and platform doors are enabled for opening, based on vital localization (with regards to the track platforms) and kinematic conditions.*

The quoted requirement addresses the train control functionality that allows the system to automatically open/close both the train and platform doors, under certain conditions (e.g., vital localization, kinematic conditions).

Then, the requirement is further specified by adopting a contract-based approach. We firstly identify the assumptions from the original text:

a_1 Valid and defined kinematic conditions;

a_2 Valid and defined vital train localization;

a_3 List of platforms described by their position on track, and the position of each platform door.

Moreover, we identify the guarantees. For the sake of brevity, we intentionally combine functional with non-functional properties in the guarantees specification. However, to properly deal with non-functional properties, two types of contracts and views are needed. We omit further details because they are out of the scope of this work.

In Guarantee g_1 and Guarantee g_2, timing specifies the maximum value of timing for which a datum remains valid. After the deadline, validity of the datum is no longer ensured; for safety reasons, it should re-calculated and required again.

g_1 Determine which **train doors** are enabled for opening. The validity duration of this value is set to 1200 msec. Undefined values shall be interpreted as not enabled;

g_2 Determine which **platform doors** are enabled for opening. The validity duration of this value is set to 1200 msec. Undefined values shall be interpreted as not enabled.

Finally, we introduced two contracts:

$$C_1 = \{a_1, a_2, a_4; g_1\} \text{ and } C_2 = \{a_1, a_2, a_3; g_2\}.$$

We model contracts in a SysML environment as follows. We deploy guarantees and assumptions to the ports of a component and contracts to the element (Figure 3). Moreover, we identify the 'constraint' UML model element to specify guarantee, assumption and contracts. Our choice is founded on two principles: to be able to deploy more than one guarantee (resp. assumption) on the same model element, and to easily access them, using the graphical facilities of the Papyrus tool.

We specify the remaining requirements via a contract-based design. We discover that some requirements are not directly refined from the top-level requirement; instead, they derive from the safety analysis (Preliminary Safety Analysis and System Hazard Analysis) and they are introduced to mitigate, or avoid, possible accidents. We trace them with suitable contracts.

Figure 3 traces two types of contracts:

- Functional contracts (graphically the blue boxes, which are highlighted with numbers from 1 to 6), which describe the functional behavior; and
- Safety contract (graphically the red boxes, which are highlighted with numbers from 7 to 9) which represents safety barriers.

Our investigation shows that functional contracts are directly derived from the top-level requirement [Req], previously quoted. However, this is not the case of safety contracts. Although this latter specifies [Req] further, it is not directly derived from [Req]. It refines a safety requirement, which has been firstly identified, secondly studied and analyzed, and, then, required to be introduced in the design specification, by the safety engineer teams to ensure the safety integrity level entailed by the CPS.

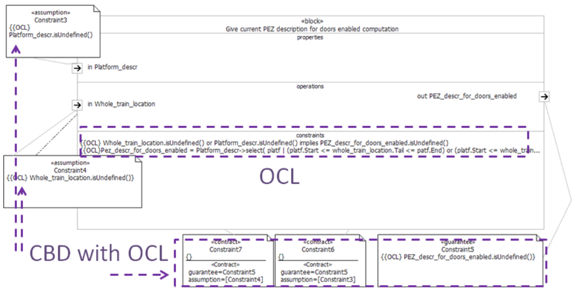
«assumption»
Constraint3
{{OCL} Platform_descr.isUndefined()
«block»
Give current PEZ description for doors enabled computation
properties
in Platform_descr
operations
in Whole_train_location
out PEZ_descr_for_doors_enabled
constraints
{{OCL} Whole_train_location.isUndefined() or Platform_descr.isUndefined() implies PEZ_descr_for_doors_enabled.isUndefined()
{{OCL}Pez_descr_for_doors_enabled = Platform_descr->select(platf | (platf.Start <= whole_train_location.Tail <= patf.End) or (platf.Start <= whole_train...
«assumption»
Constraint4
{{OCL} Whole_train_location.isUndefined()}
OCL
CBD with OCL
«contract»
Constraint7
{}
«Contract»
guarantee=Constraint5
assumption=[Constraint4]
«contract»
Constraint6
{}
«Contract»
guarantee=Constraint5
assumption=[Constraint3]
«guarantee»
Constraint5
{{OCL} PEZ_descr_for_doors_enabled.isUndefined()}

Figure 4: Contract-based approach to Model-Based system engineering

The (red and Number 8) contract has a means to highlight traceability of safety requirements, which are previously captured by the safety engineers teams during the Hazard Analysis at the early stage of the system development.

At the meta-modeling level, we then introduce Stereotype 'MitigationContrats' that has the primary role to trace the link between a contract at design space and the original specification at safety space.

Figure 4 shows a comparison between requirements specified via contract-based approaches, and requirements specified with a textual flat language. We intentionally adopt the same formal language: the international OMG standard 'Object Constraint Language' (OCL) [24], which is compliant with SysML and hence the two standards can be easily applied together to the same model. OCL is a formal language that allows engineers to specify requirements or more in general constraints, thanks to the help of a formal syntax, in a model previously specified (for example by UML, SysML, MARTE).

Figure 4 shows two contracts: they have the same guarantee, but differ from the assumptions. The assumptions and guarantee are clearly deployed on the related model elements and are correlated via a contract.

The block includes an OCL constraint, specified in the usual manner. The constraint has the following form $A \vee B \rightarrow C$, where A and B correspond to the previous assumptions and C to the guarantee. However, such a flat formulation does not clearly highlight the association between the atomic formula (A, B or C) and the model element; the only way we have to recognize such a correspondence is by the name (for example, Whole_train_location.isUndefined() in the formula corresponds to the Port with name Whole_train_location).

An advantage in the use of contract-based approaches is to structure the link between an OCL atomic formula and the corresponding model element.

4.1 Preliminary Feedback

During this work, we have been able to compare CBSE with the textual requirements approach and CBSE with the textual contracts approach. Even if the expressive power remains equivalent, contracts have the advantage to drive the component breakdown structure analysis and design by facilitating

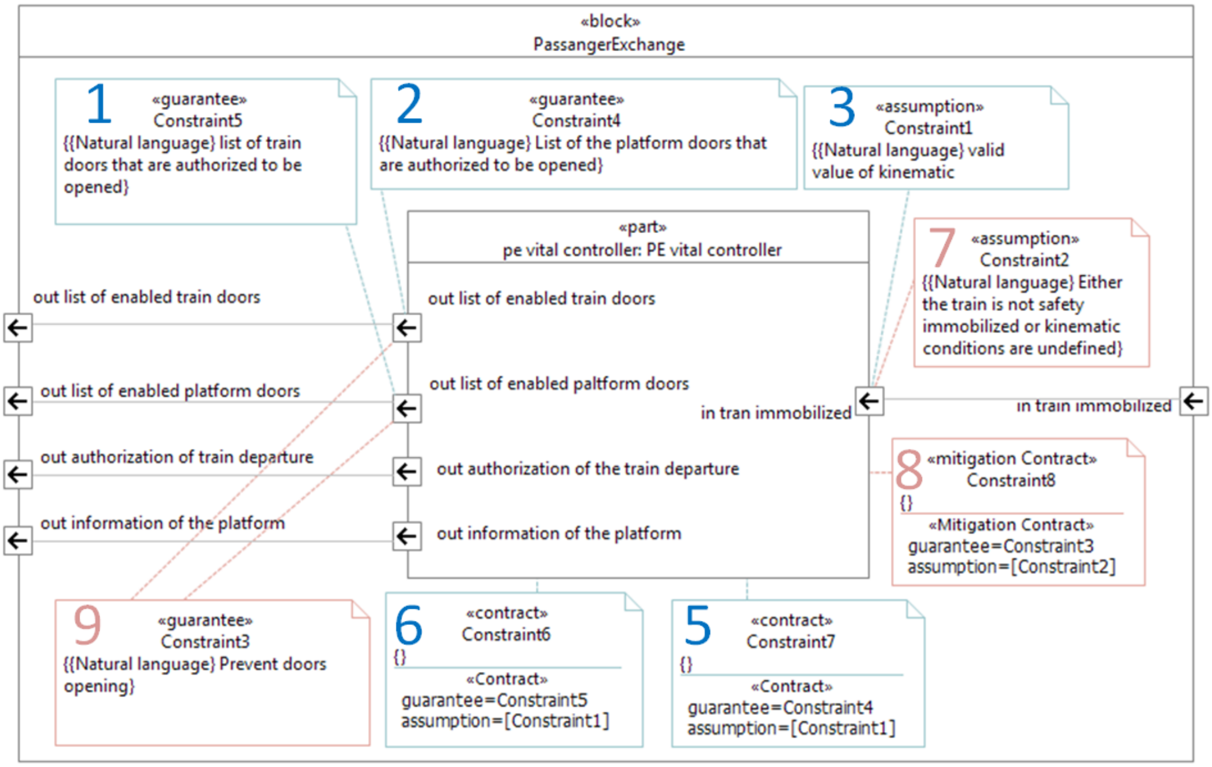
«block»
PassangerExchange
1 «guarantee» Constraint5
{{Natural language} list of train doors that are authorized to be opened}
2 «guarantee» Constraint4
{{Natural language} List of the platform doors that are authorized to be opened}
3 «assumption» Constraint1
{{Natural language} valid value of kinematic
«part»
pe vital controller: PE vital controller
7 «assumption» Constraint2
{{Natural language} Either the train is not safety immobilized or kinematic conditions are undefined}
out list of enabled train doors
out list of enabled train doors
out list of enabled platform doors
out list of enabled paltform doors
in tran immobilized
in train immobilized
out authorization of train departure
out authorization of the train departure
out information of the platform
out information of the platform
8 «mitigation Contract» Constraint8
{}
«Mitigation Contract»
guarantee=Constraint3
assumption=[Constraint2]
9 «guarantee» Constraint3
{{Natural language} Prevent doors opening}
6 «contract» Constraint6
{}
«Contract»
guarantee=Constraint5
assumption=[Constraint1]
5 «contract» Constraint7
{}
«Contract»
guarantee=Constraint4
assumption=[Constraint1]

Figure 3: Contract-based approach to Model-Based system engineering

the allocation and refinement of functional and safety behaviours on sub-components. It seems also a promising mean for structuring verification and validation activities. Finally, thanks to their inherent ability for traceability, contracts are good candidates to strengthen a development process compliant with CENELEC norms.

5 Conclusion and On-Going Work

In this position paper, we introduce the overall view we pursue to deal with seamless guarantee of safety-related properties from CPS design to execution platform in the FSF project [9]. The vision outlined exploits contracts as a means to identify precisely the conditions for a correct interaction of components as well as to specify which assumption a functional level (code) should require to a hardware level to ensure the acceptable threshold of SIL. Although our work is at an early stage of development, we feel that this approach can provide a simple, but firm, foundation to a rigorous approach for reasoning on the interaction of safety-related properties in CPS.

6 Acknowledgment

This research work has been carried out under the leadership of the Technological Research Institute SystemX [25], and therefore granted with public funds within the scope of the French Program *'Investissements d'Avenir'*. This work was also supported in part by the CyPhERS FP7 project, grant agreement no. 611430.

References

[1] C. Ptolemaeus, ed., *System Design, Modeling, and Simulation using Ptolemy II*. Ptolemy.org, 2014.

[2] L. de Alfaro and T. A. Henzinger, "Interface automata," in *Proceedings of the Ninth Annual Symposium on Foundations of Software Engineering*, pp. 109–120, ACM Press, 2001.

[3] L. Benvenuti, A. Ferrari, L. Mangeruca, E. Mazzi, R. Passerone, and C. Sofronis, "A contract-based formalism for the specification of heterogeneous systems," in *Proceedings of the Forum on Specification, Verification and Design Languages*, FDL08, (Stuttgart, Germany), pp. 142–147, September 23–25, 2008.

[4] R. Passerone, I. B. Hafaiedh, S. Graf, A. Benveniste, D. Cancila, A. Cuccuru, S. Gérard, F. Terrier, W. Damm, A. Ferrari, L. Mangeruca, B. Josko, T. Peikenkamp, and A. Sangiovanni-Vincentelli, "Metamodels in Europe: Languages, tools, and applications," *IEEE Design and Test of Computers*, vol. 26, pp. 38–53, May/June 2009.

[5] D. Cancila, R. Passerone, T. Vardanega, and M. Panunzio, "Toward Correctness in the Specification and Handling of Non-Functional Attributes of High-Integrity Real-Time Embedded Systems," *IEEE Transactions on Industrial Informatics*, May 2010.

[6] A. Sangiovanni-Vincentelli, W. Damm, and R. Passerone, "Taming Dr. Frankenstein: Contract-Based Design for Cyber-Physical Systems," *European Journal of Control*, vol. 3, pp. 217–238, 2012.

[7] P. Derler, E. A. Lee, M. Torngren, and S. Tripakis, "Cyber-physical system design contracts," in *International Conference on Cyber-Physical Systems (ICCPS 2013*, (Philadelphia , USA), 2013.

[8] CyPhERS FP7 Project, "Cyber-Physical European Roadmap and Strategy." `http://cyphers.eu/`.

[9] FSF IRT SystemX Project, "Fiabilité et Sûrêté de Fonctionnement (Reliability and Safety)." `http://www.irt-systemx.fr/wp-content/uploads/2013/03/FiabiliteetSuretedeFonctionnement.pdf`.

[10] CENELEC, "Railway applications - The Specification and Demonstration of Reliability, Availability, Maintainability and Safety (RAMS) - Part 2: Systems approach to safety." CENELEC standard `http://www.cenelec.eu/`, 2012.

[11] ALSTOM, "Metropolis And Metro Train Solution." `http://www.alstom.com/`.

[12] IEEE, "IEEE Standard for Communication Based Train Control Performance Requirements and Functional Requirements." IEEE standard. `http://standards.ieee.org/`, 1999.

[13] ALSTOM, "Alstom ASAP methodology: Advanced System Architect Program." OMG `http://www.omgwiki.org/MBSE/doku.php?id=mbse:alstomasap`.

[14] Marco Ferrogalini, Jean Le Bastard, "Return of experience on the implementation of the System Engineering approach in Alstom." OMG `http://www.omgwiki.org/MBSE/lib/exe/fetch.php?media=mbse:rex\_on\_se\_approach\_implementation\_in\_alstom.pdf`.

[15] ASSERT FP6 Project, "Automated proof-based System and Software Engineering for Real-Time systems project." `http://www.assert-project.net`.

[16] CHESS Project, "Composition with Guarantees for High-Integrity Embedded Software Components Assembly." `http://www.chess-project.org`.

[17] J. Sifakis, "Embedded Systems - Challenges and Work Directions," in *Principles of Distributed Systems* (LNCS, ed.), vol. 3544, 2005.

[18] W. Damm, H. Hungar, B. Josko, T. Peikenkamp, and I. Stierand, "Using Contract-based Component Specifications for Virtual Integration Testing and Architecture Design," in *Proceedings of DATE conference*, pp. 109–120, 2011.

[19] AUTOSAR, "Automotive Open System Architecture." `www.autosar.org`.

[20] A. Pinto, A. Bonivento, A. L. Sangiovanni-Vincentelli, R. Passerone, and M. Sgroi, "System level design paradigms: Platform-based design and communication synthesis," *ACM Transactions on Design Automation of Electronic Systems*, vol. 11, pp. 537–563, July 2006.

[21] A. L. Sangiovanni-Vincentelli, "Quo Vadis, SLD? Reasoning About the Trends and Challenges of System Level Design," *Proceedings of the IEEE*, vol. 95, pp. 467–506, March 2007.

[22] A. Davare, D. Densmore, L. Guo, R. Passerone, A. L. Sangiovanni-Vincentelli, A. Simalatsar, and Q. Zhu, "METROII: A design environment for cyber-physical systems," *ACM Transactions on Embedded Computing Systems*, vol. 12, pp. 49:1–49:31, March 2013.

[23] D. Densmore, R. Passerone, and A. L. Sangiovanni-Vincentelli, "A platform-based taxonomy for ESL design," *IEEE Design and Test of Computers*, vol. 23, pp. 359–374, May 2006.

[24] OMG, "Object Constraint Language (OCL)." OMG standard.`http://www.omg.org/spec/OCL/`.

[25] IRT, "Institut de Recherche Technologique (Technological Research Institute)." `http://www.irt-systemx.fr/`.