Reducing the Gap Between Formal and Informal Worlds in Automotive Safety-Critical Systems

HugoG. Chalé, Ofaina Taofifenua,
Thierry Gaudré, Alexandra Topa
RENAULT
1 avenue du Golf
78288 Guyancourt, France

Nicole Lévy
Laboratoire CEDRIC, CNAM
292 Rue St Martin
75141 Cedex 03, France

Jean-Louis Boulanger
CERTIFIER
1, place Boussu BP70141
59416 Anzin Cedex, France

Abstract. The upcoming ISO26262 standard, which deals with the functional safety of road vehicles, will induce car manufacturers to adapt the way in which vehicle systems are usually developed. To achieve this, more rigorous development processes along with new tools and techniques will most certainly be necessary. This paper presents an overview of current initiatives at Renault dealing with the improvement of development processes for mechatronic systems to comply with ISO 26262. It focuses on introducing more formalization in the systems engineering design process via the definition of an ontology to formalize the concepts and knowledge of the systems engineering, functional safety and automotive specialty domains (*e.g.* braking, energy management). The ontology is at the heart of our improvement initiatives since it allows establishing logical consistency of the whole design process. A regenerative hybrid braking system integrated into a full electrical vehicle will serve as the case study for the evaluation of the improvements made possible by the approach.

Introduction

The Context. Car manufacturers have had to face an always-increasing list of stakes and challenges for the last several years. The strongly competitive worldwide market of today imposes a car manufacturer to offer to its customers relevant, innovative, reliable, environment-friendly and safe services, at competitive costs while complying with ever more stringent regulations and times-to-market (Chalé Góngora *et al* 2010). The solutions to face this challenges have typically been the development of mechatronic systems, which make an integrated use of mechanical, electronic and software technologies (Bishop 2008), and the implementation of model-based development processes (Struss and Price 2004).

This causes an increase in system complexity, which makes safety analyses on these systems equally more complex, time consuming and thus more expensive, which is incompatible with the economic constraints and the relatively short development cycles of the automotive industry. Mastering safety risks is nevertheless necessary, as illustrated by the many examples of car manufacturers forced to perform important vehicle recalls because of failures that are not

always immediately identified or well understood. Furthermore, the arrival of the ISO 26262[1] standard (ISO 2009) regarding the functional safety of electrical electronic (EE) embedded systems brings along new requirements with which automotive systems and development processes will have to comply. Although the standard will not be compulsory at a first stage, it is already acting as a catalyst for the research of improved processes, methods and tools in order to master safety risks.

Motivation. One of the current challenges at Renault consists in preparing its engineering divisions so that they are capable of developing mechatronic safety-critical systems according to the ISO 2626 standard. The standard defines a system life cycle and the activities that must be performed in the different phases of this life cycle, along with the support processes that are necessary for these activities. It also defines a specific method for automotive hazard analysis that identifies hazards and classifies them using ASIL[2], for Automotive Safety Integrity Levels. The result of this analysis is the definition of couples {Hazard, ASIL} called Safety Goals. Safety goals are allocated from the system level to its components according to the rules defined by the standard. This leads to the definition of specific safety requirements on the system, on its components and on the associated development processes, depending on the ASIL quotation. The satisfaction of these requirements allows asserting the absence of unacceptable residual risks.

Therefore, the standard raises some problems concerning the demonstration of functional safety and, more generally, concerning the development processes which are currently under-formalized. Indeed, one of the strengths of ISO 26262 is that each requirement in the standard is associated to an ASIL. So, the compliance of the system, of its components (whatever their nature) and of their development processes to the standard can be obtained and verified in a systematic way. This suggests that better formalization can be beneficial to ensure consistency with respect to the standard.

This paper presents an answer to this state of affair. We build upon our previous work (Chalé Góngora *et al* 2010) and propose to introduce the use of ontology into the current systems development process at Renault. A regenerative hybrid braking system was chosen as our case study in order to evaluate the foreseen techniques on a non-trivial real application. The advantages of the proposed ontology-centric approach are:

- A formal definition of all the concepts and relationships (our system and safety design language or data model) shared and understood by all actors from different domains.
- The possibility to verify that all the data describing a system (*e.g.*, requirements, functions, flows, components, interfaces, etc.) are consistent and complete.
- Eased impact analyses following a change request and eased modifications of all impacted data.
- The existence of numerous tools supporting the manipulation of the ontology.

In the first part of the paper, we present the current design process at Renault, point out some of its limitations and propose solutions to overcome them. We then introduce our ontology and illustrate the role it could play in a development process. Next, we present the link between the ontology and the work that has been done on model transformation. Finally, we conclude on

[1] ISO 26262, currently under final revision, is the automotive adaptation of IEC 61508, an international generic standard on the functional safety of electric, electronic and programmable electronic safety related systems. Its generic scope , which has made it a reference for all the main industrial sectors, and has made IEC 61508 the object of numerous adaptations that take into account the specificities of these different sectors.

[2] ASIL are defined in four levels of requirements and measures of conformity that allow asserting the absence of an unacceptable residual risk: A, B, C and D, the latter representing the most stringent level.

the future perspectives that are envisaged to further promote formalization in Systems Engineering (SE) development process at Renault.

Current Design Process at Renault

The development processes of automotive systems practiced by car manufacturers or by their suppliers are continuously evolving in order to make them either more competitive or compliant to the regulations that are applicable to the automotive industry. The work presented here focuses on the specification and design phases of the development process (i.e. the descending branch of the "V" cycle), since it is during these phases that stakeholders and customer requirements are elicited, transformed and allocated to system components that will be developed by the suppliers of the car manufacturer.

The System Design Process. Figure 1 presents a simplified illustration of the system design process.

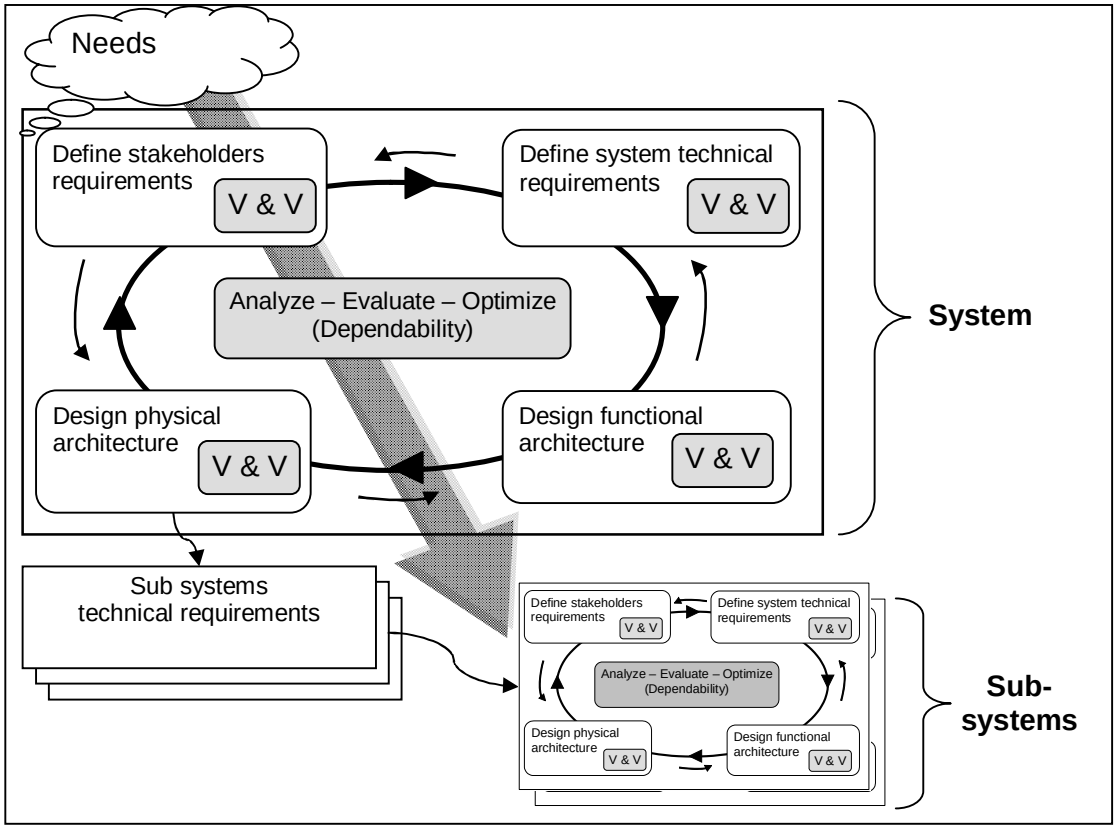

Figure 1. System design process at Renault

The four main activities of this design process are the elicitation of stakeholders requirements (capture and reformulation of original needs), the definition of system technical requirements, the functional architecture design and the physical architecture design. Let us introduce the regenerative hybrid braking system (RCB for Regenerative Combined Braking) case study to illustrate this process. This system is integrated into a full electrical vehicle and combines basic hydraulic and advanced electro-mechanical technologies for braking actuators.

One of the missions of this system is to recuperate as much electrical energy as possible by using the electric motor of the powertrain and stock this energy for future use, when the driver brakes or decelerates the vehicle. During the elicitation of stakeholders requirements, the

original stakeholders needs are captured in the "System Stakeholder Requirements document" (SSR), which contains requirement statements that are not always clear or precise. For example, one of the requirements of the "braking service" stakeholder states: "*The RCB system shall allow the driver to modulate the deceleration of the vehicle in a stable way*". When defining the system technical requirements, stakeholder requirements are transformed into a set of precise, achievable and consistent requirements, with tradeoffs between incompatible stakeholder requirements being possible. For example, one technical requirement associated to the stakeholder requirement cited above states: "*The RCB system shall ensure the stability of the vehicle during braking by taking into consideration the vehicle mass, the wear and temperature of the braking pads and the environmental driving conditions*". Technical requirements are gathered in the document called "System Technical Requirements document" (STR).

In functional architecture design, the functions or transformations that the system must perform in order to satisfy the requirements stated in the STR (mainly functional requirements) are identified, described and made precise (decomposed). The flows (*i.e.*, information, energy or material flows) that are used (produced or consumed) by the functions are specified as well. The internal behavior of the system is also described and corresponds to the logic execution of the functions of the system. During physical architecture design, we define the constituents of the system, their interfaces and their connections to fully satisfy the technical requirements (mainly the non-functional ones like cost, weight, size, forbidden or authorized use of materials, *etc.*). During this process, the functions of the system are eventually decomposed to be allocated on the constituents and the flows are associated with the interfaces and connectors that transport them. These architectures are usually portrayed in the form of bloc-diagram type models. For reasons of confidentiality, the system architecture (functional and physical) is not presented in this paper.

The activities related to system dependability are represented in the center of figure 1 because they take place during all the activities presented above. We focus on the safety aspects of dependability as the main objective is to comply with ISO 26262 standard. The activities start as soon as the stakeholders requirements have been specified. The main activities are, on the one hand, the Preliminary Hazard Analysis (PHA), which lists and evaluates the hazardous events of the system (Feared System Event (FSE), from the system viewpoint, and Feared Customer Event (FCE), from the customer point of view) in order to define the safety goals at the system level. And, on the other hand, the analysis of the causes leading to the hazardous events. This cause analysis will enable to derive the safety goals on the system into safety requirements on the functional and physical architectures and on the elements of these architectures.

All the above-mentioned activities are the object of verification and validation activities (V&V in the figure) such as documents inspections, simulation, impact analysis of change requests, *etc.* Those processes are iterative and form a design loop that can be repeated partially or completely following the evolution of needs, the emergence of constraints or the impossibility to realize some constituents (*i.e.*, sub-systems) of the system under development. The sub-systems are then developed following either a similar process to the one we presented, if the sub-system is complex in the sense that it calls for different professional fields, or a specific process, if the subsystem calls for only one professional field such as software, electronics, mechanics, *etc.*

Problems of the Current Process. The development of automotive mechatronic systems requires the participation of different professional fields (*e.g.*, vehicle architecture, mechanic, electronic, software, *etc.*), each having its own language, its own jargon. Knowledge and information are often implicit to one specific professional field. They are known to experts or specialists of the profession, but are not always well capitalized and, therefore, they are unknown to the other fields or, even worse, they might be lost if those experts or specialists change of position. It is the role of the system engineers to effectively take into account all those system stakeholders (*i.e.*, the professional fields concerned with the system) and orchestrate their contributions in the big picture as to develop a correct system solution. In other words, system engineers must overcome a *consistency problem*.

From a syntactic point of view, the consequences are not too severe. Syntax consistency problems arise when two different terms are used to name one same thing. As a usual example, we often work with documents and models that have terms in English and French languages. Consequently, we might say that working with two models with different names "just" takes more time. As the meaning is not altered, we can somehow understand how it all comes down together. It is just a matter of realizing that a given actor calls "this thing that way" and living with that. From the semantic point of view, however, the problem takes a completely different dimension. The problem can be summarized as the utilization of one same term by two different professional fields to designate respectively two different concepts. This can lead to situations that are so contradictory that we might end up trying to solve a problem with no solution. Ultimately, when the actors of different professional fields exchange information or knowledge, some content is lost either by communication omission or by misinterpretation of this information as mentioned by (Burr *et al* 2005). The other possible consistency problem is less fundamental but equally important and consists in inter-domain consistency. As the system development is carried out by different domains, each of them relies only on the information relevant to their activity. The information manipulated by the different domains can intersect and the difficulty is to guarantee that all the domains are working with consistent information ensuring consistency of the design process (Papadopoulos *et al* 2001).

Transition to Model-Based System Engineering (MBSE). Renault is currently transitioning to a MBSE process for the development of its vehicle systems. The use of formal and informal (but consistent) models to create a common semantic model is expected to facilitate systems engineering activities and to avoid the encountered drawbacks of previous document-centric implementations of the process, which were lacking semantic consistency among the different modeled objects. The objective of MBSE is to produce and control a consistent, correct and complete global model of the system, which contains all the information that specifies, designs or will allow the verification and validation of the system. The main benefits of implementing MBSE, as they are emphasized in (Estefan 2008 and Friedenthal *et al* 2008), include:

- improved quality through a more rigorous and costless traceability between requirements, design, analysis and testing
- increased productivity through the reuse of models and automated document generation
- enhanced communication by integrating views of the system from multiple perspectives

The risk of developing inconsistent models that have different conceptualizations of the same system according to their own viewpoint still remains very present. Inconsistencies between models discovered too late in the development process may produce huge costs. Consistency is then a crucial issue and needs to be maintained at all levels in the development process. Therefore, the consistency problem concerning MBSE can be formulated as the demonstration of the consistency of two different designs models.

We have opted to build an automotive ontology because of its capability to describe in a formal and explicit way the concepts of a domain, their properties and their relationships as it seems essential to facilitate the communication between the actors of a project (shared common language, minimization of information loss, improvement of information capitalization and reuse). The interested reader can refer to Grubber (1995) to understand the concepts upon which an ontology is built upon to develop those capabilities. Another key element in MBSE is the transformation of models, which allows the definition and implementation of operations on models. This provides a transformation chain that enables the automated or computer assisted development of a system from its corresponding models. Model transformations implicitly embed some semantic knowledge that ensures the inter-model consistency. In the next section, we present an ontology as a system consistency reference model placed at the heart of the system design process.

Ontology Centric Design Process

As shown in figure 2, we propose to introduce an ontology as the central element of the system design process. In this figure, we separate into two branches the activities pertaining to system design and safety presented in the previous section. The ontology is instantiated for the system under development. It will serve as the consistency reference model in the RCB project, which follows a model-based approach presented in (Chalé Góngora *et al* 2010).

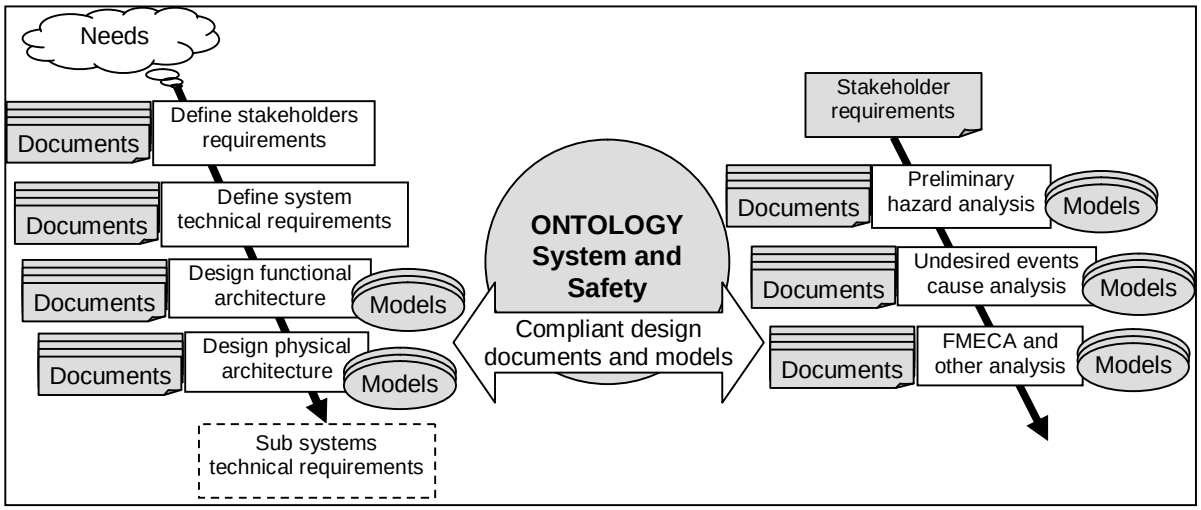

Figure 2. Central role of the system and safety ontology in the design approach

Use of the Reference Model. The actors of a development project, independently of their respective fields or area of expertise, will refer to the ontology (a shared conceptualization of the system and safety engineering domain and of the system under development) to verify and validate the compliance, the completeness and the consistency of the information (*i.e.*, documents and models) produced by the system design and safety activities.

Figure 3 illustrates some possible uses of an ontology in a model-based approach. The figure presents the example of two Simulink models, but the approach is applicable to other types of models. In this example, we are interested in the signals (*i.e.*, the solid straight arrows) of the Simulink models. The ontology models this concept with the *flow* class with an attribute (not represented in the figure) maxValue. We can define semantic consistency relations with the help of transformations or mappings between the domains of the ontology and of the language of the Simulink tool, on the one hand, and between the instances of the ontology and the instances of the elements of Simulink, on the other hand.

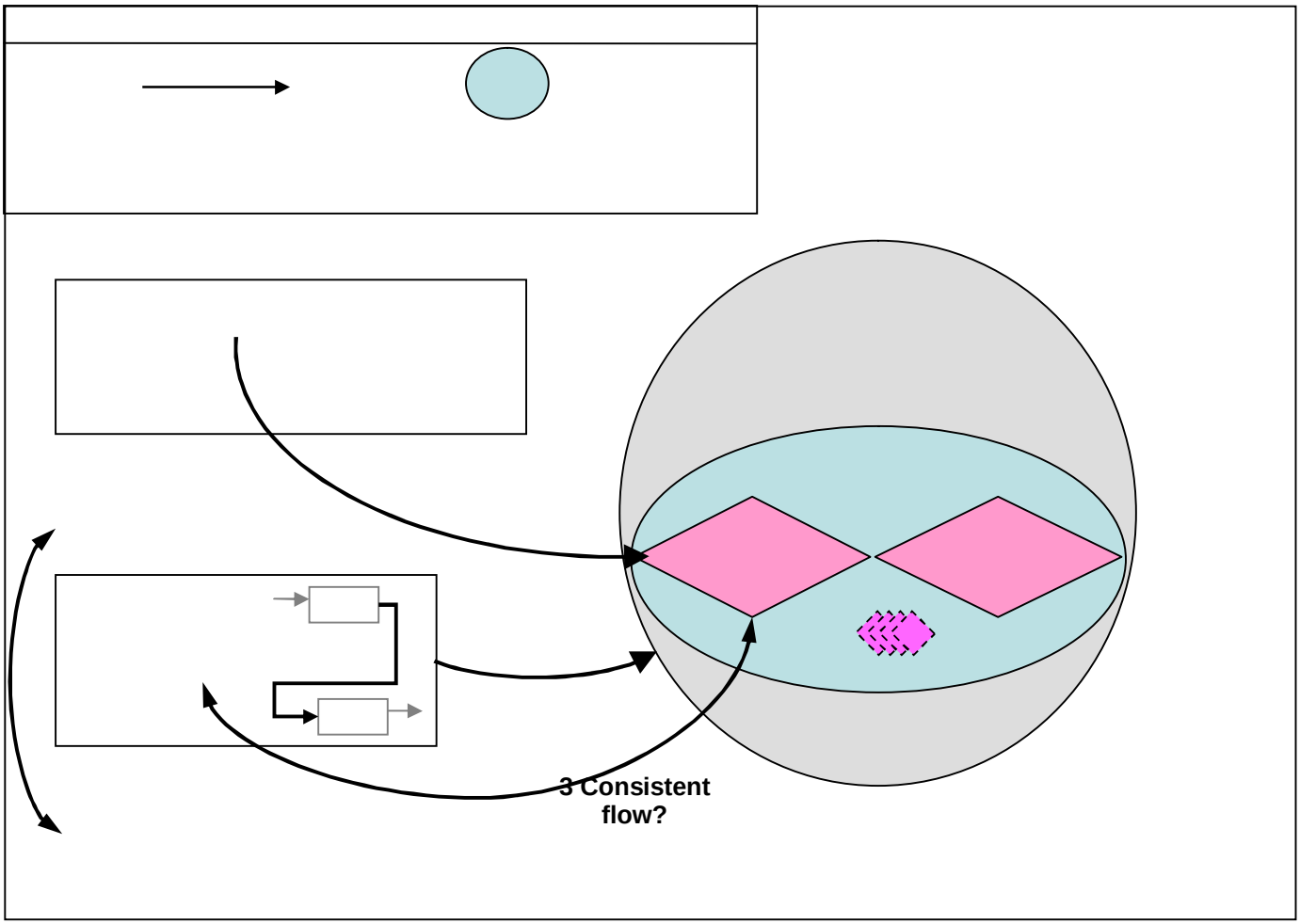

Figure 3. Uses of an ontology as a reference model of a MBSE approach

Assuming we have defined that Simulink signals are equivalent to the ontology flows, it is then possible to:

1. Enrich the ontology: All the signals of a Simulink model will enrich the instances of the ontology. In the figure, the signal *Torque_Frein_Electrique* of the Simulink model defines the flow *Flow_001* in the ontology. For this flow, we define a unique maximal value of the braking force *maxTorque*.
2. Use knowledge in the ontology: A second Simulink model will be able to use the flows of the ontology and gather the information previously defined. In the figure, the flow *Flow_001* of the ontology and the signal *Electrical_Brake_Torque* of the second model are equivalent. In Simulink, this signal should connect to a port that enables to type the flow. In our example, this value has an upper bound equivalent to *maxTorque*.
3. Verify the consistency of a model with respect to the ontology: If the signal *Electric_brake_Torque* (that models a flow representing the braking torque of the electrical engine) does not exceed in simulation the maximum value maxTorque, then the signal is coherent to the ontology (for the maxValue relation). Generalizing to all the relations defined into the ontology, we can assess the consistency of a model compared to the ontology.
4. Verify that two models are consistent: If we had defined a mapping between *Flow_001* and *Torque_Frein_Electrique* and between *Flow_002* and *Electrical_Brake_Torque*,

then a user can notice that those two instances are equivalent since they represent the same element in the system, even though two designations are used in the models (one in French and the other in English). Defining an equivalence between those two instances will result in an inconsistency. In figure 3, two different values for the maxValue have been defined (i.e., *maxTorque* is different from *maxTorque2*) whereas in the ontology we specified that a flow can only have one maxValue. In the opposite case (i.e., *maxTorque* is equal to *maxTorque2*) and generalizing, the two models are consistent and, once again, they describe the same system and we have some evidences that a solution for the system exist. Generalizing even more, it becomes possible to verify the consistency of the whole system design through an ontology.

In a general manner, the project actors create information in documents and models. The ontology will enable them to verify the consistency, the completeness and the conformity of the produced information. Once verified, the new information can be imported into the ontology (Kergosien *et al* 2010). Another essential use of the ontology, not presented is this paper, is the possibility to query the ontology as a knowledge base and infer knowledge. Reference information can be exploited to produce new views (system views) and generate new information (Sure *et al* 2002). The ontology is therefore the reference (model) that, on the one hand, contains the reference information that describes the system under development and, on the other hand, connects the information it contains with the information present in the documents and models produced during the course of the system development project.

The System and Safety Ontology. Figure 4 shows a part of the system and safety ontology under development. As presented in (Chalé Góngora *et al* 2010) the ontology serves as a data model for the systems covered by the ISO 26262 standard. This data model formalizes the relevant concepts of the systems engineering domain, as defined by INCOSE (*e.g.*, requirements, functions, *etc.*), and of the safety domain, as defined in (Avizienis *et al* 2004) and in the ISO 26262 standard (*e.g.*, safety goal, safety requirement, ASIL, *etc.*), as well as the relations between all those concepts.

We use the "Ontology Web language" (OWL 1.0[3]), a formal language with mathematically defined syntax and semantics[4]. For instance, a *Safety Goal* is defined by a couple *Feared System Event* (FSE) and *ASIL*. Defining a safety goal that is in relation with two different ASILs would then result in a contradiction on the cardinality restriction between safety goal and ASIL. The ontology is realized with the help of Protégé 3.4.4[5]. This version of Protégé includes in particular a plug-in for the "Semantic Web Rule Language" (SWRL) and an interface with the rule engine Jess[6]. This plug-in allows expressing rules on the ontology that can be used to infer more complex knowledge than with the use of OWL only. For instance, it is possible to express the logical consequence that, if a requirement is implemented by a function and the requirement is part of another requirement, then this other requirement is also implemented by the function. This enables to show, for example, all the functions that participate to the satisfaction of a requirement.

[3] www.w3.org/2004/OWL/

[4] Attention must be paid to the sense of semantics. It just allows to *describe* the structure of the universe of discourse and by no means tries to explain the universe of discourse.

[5] protege.stanford.edu/

[6] protege.cim3.net/cgi-bin/wiki.pl ?SWRLTab/

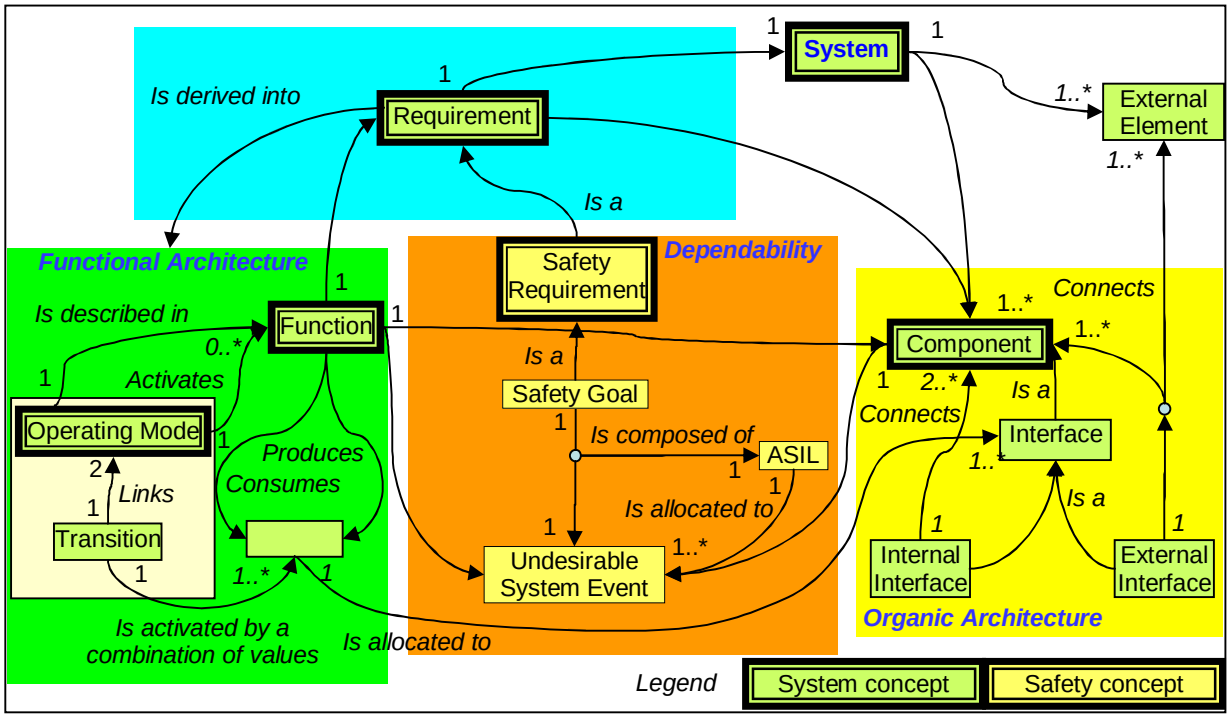

Figure 4. System and safety ontology

As of now, the ontology consists in 97 classes, *i.e.*, concepts, and more than 100 properties, *i.e.*, relations. In figure 4, we present only the most general concepts and relations. The instantiation of the ontology has started on a subset of the case study for demonstration purposes. It results that the reference model of the RCB system comprises 66 requirements, 79 functions with 83 flows and 22 components (not including interfaces).

In the next section, we discuss about model-driven approaches and model transformations and explain how they can be integrated in the framework of the ontology centric design process.

Model transformation

Applying Model-Driven Architecture to System Engineering Processes. Model-driven engineering (MDE) or Model-driven development (MDD) is originally a discipline in software engineering that relies on models, but introduces a higher level of abstraction by defining meta-models as first class entities. The idea behind MDD is to create different models of a system at different levels of abstraction, in order to achieve an architectural separation of concerns, and to use transformations to produce the desired implementation. In current literature, we can find various approaches to model transformation techniques. The best-known MDD realization is the MDA approach of the OMG who introduced this architectural framework in order to perform correct and automatic model transformations that provide increasing capabilities regarding costs, quality and delivery cycles challenges. Numerous recent efforts and studies investigate the applicability of MDA principles to the Systems Engineering domain. A 10-20% efficiency gain is expected once this approach is applied. Before that, industrial companies have to understand this approach, adapt it to SE domain and adapt their organization as well. In (Estefan 2008) it is highlighted how OMG MDA is applied to a typical SE life cycle, taking into account the artifacts and the deliverables associated with each MDA view. Tools provided to support MDA in a MBSE approach are expected to become sufficiently mature in the near future with automated transformation processes.

Model transformation is an essential part of the MDA framework. In this framework, models are based on meta-models that comply with the Meta-Object Facility (MOF) standard of the

OMG that uses the layered concepts of instance, model, meta-model and meta-meta-model. Model transformation is the automatic generation of a target model (the result of the transformation) from a source model (the input of the transformation) by a transformation engine according to a transformation model (a set of transformation rules), see figure 4 below.

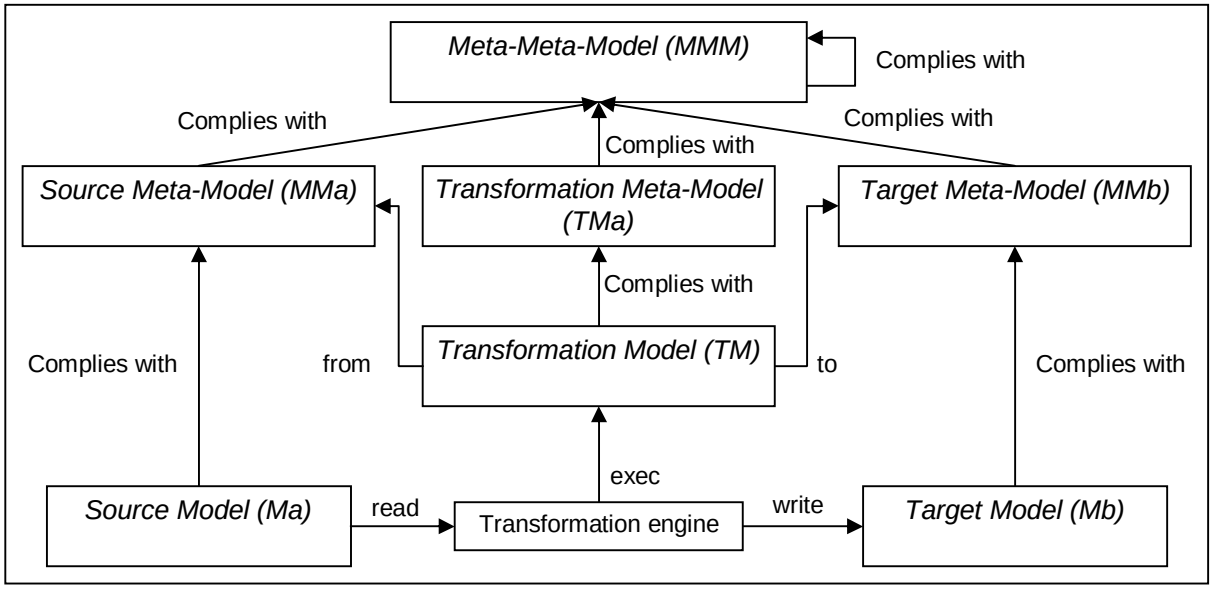

Figure 5. Elements of model transformation

Transformation rules are defined whenever possible for the meta-model level and written as expressions of transformation languages. In the MDA framework, transformation rules are entered into a transformation tool, which can then automatically interpret them and execute the transformation. For that purpose, a formal syntax for writing transformation rules must be defined (Anneke *et al* 2003). In the case of automatic model transformations, the mapping between the different concepts has to be developed only once for a pair of meta-models, not for each model instance (Levendovszky *et al* 2002). Therefore, the specification of meta-models is a prerequisite for the execution of automatic model transformation.

Model Transformation Panorama. As already mentioned, a major advantage of the model based development process consists in the provided support for the analysis and the construction of a consistent, correct and complete system model. In order to produce a coherent global system model integrating different views of the system at different stages of the development process and at different abstraction levels, two elements are important:

- the techniques used to perform analyses on the models along with the establishment of traceability,
- the languages used by those techniques.

Modeling languages must have sufficient power to impose consistent modeling rules and to allow automatic (i.e. tool supported) analyses and transformations through a non-ambiguous interpretation of model elements. The final goal is to have a complete and seamless system and component development environment (methods, techniques, tools) which supports the whole system and components design process regarding safety aspects. To achieve this goal, one future perspective is to analyze all the languages (general or domain specific) in order to identify the model transformations that are necessary to obtain a chain of model transformations enabling the automated or computer assisted development of the system. Further investigations must be concentrated on choosing the most appropriate and efficient transformations covering all activities within the adopted process inside Renault.

An intermediate result is presented in table 1. This table shows a partial first comprehensive panorama on existing and required model transformations, which will be able to cover the Renault SE process currently under deployment. We performed a study that focused on existing transformation approaches (adopted techniques and defined transformation rules). Some approaches are strengthened by implemented tools that serve as a proof of concept, some are commercialized and others have yet to be implemented.

Table 1: Model transformation panorama (excerpt)

Source Language	Target Language	Artifacts to be transformed (source/target)	Existing Transformation Tool/Provider or Specified Transformations	Transformation Approach	Technical Constraints of Transformation Tools/Remarks
UML	B Language	**Source artifacts:** class diagram, state machine diagram **Target artifacts:** formal B specifications	**Tool:** UML2B **Tool:** UML-to-B	**UML2B:** Pattern matching through the use of Objective CAML language which facilitates defining transformations rules **UML-to-B**: MDA-based transformation technique founded by a set of structural and semantic mappings between UML and B meta-models	**UML2B:** Working on Linux, XMI standard support (especially those generated by Objecteering UML Modeler 5.2.1). The user must dispose of Objective CAML language (version 3.04 or higher) and a PXP parser **UML-to-B:** UML and B meta-models are encoded with Eclipse IDE thanks to the Eclipse Modeling Frame-work (EMF) and more precisely thanks to eCore tool (the eCore language used to create models in EMF conforming to the MOF) Transformation rules are written in oAW2 (openArchitectureWare) using xTend tool, which supports MDA
B Language	UML	**Source artifacts:** formal B specifications **Target artifacts**: class and state machine diagrams (future support of sequence and collaboration diagrams)	**Tool: B2UML**	Pattern matching	**B2UML:**automatic transformation process; it does not allow designer intervention

Ontology-Based Model Transformation Framework. In MBSE, a model allows capturing the relevant aspects of a system from a given perspective, and at a precise level of abstraction. During the system development, different model types are realized to represent specific possible system views for any of the design process activities (specifications analysis, system architectural design, validation and safety analysis). The models should contain only the aspects needed to support the design process phase they are used in, hiding unnecessary complexity. Models are supported by languages that have at least a well defined structure (*i.e.*, syntax) and in some cases a well defined meaning (*i.e.*, semantics). When the syntax and semantics are well defined (*i.e.*, mathematically defined) the language is connoted formal. In MDA, meta-models are used to define the syntax and semantic of languages. Most meta-models are semi formal in the sense that their syntax is formal but their semantics is not. Following those semi formal meta-models, model transformations operate essentially at the syntax level but always embed implicitly some semantic knowledge (Roser and Bauer 2005) that ensures the inter-model consistency. We argue that the system and safety ontology can make explicit the part of the semantic knowledge that is common to the source and target domains involved in a transformation.

We propose a framework that ensures model transformations consistency with the ontology. Figure 5 illustrates this framework. We build upon the framework of model transformation as defined by the OMG so we can see two meta-models, source and target, with the transformation model in the center. For the purpose of the example, we represented two

transformation models in the figure. At the top of the figure, we place the system and safety ontology that defines those domains with their concepts and relations. Actually, both meta-models and ontologies can be used to define concepts and relations (Söderström 2001) so meta-models and ontologies can be used independently as the meta-model in MOF for model transformations. The terms mapping and transformation are interchangeable and can be used indifferently but the term mapping is encountered more frequently in ontology literature so we will use this term when the transformation involves an ontology.

In the framework, a model transformation is still defined independently from the ontology. However, the meta-models involved in the transformation need to be mapped with the ontology, so we define one mapping from the ontology to each meta-model (*Source Mapping* and *Target Mapping)*. In the figure, we represented only some concepts of the meta-models and the ontology but the idea can and has to be generalized to relations. Mapping the ontology to the meta-models involved in a transformation enables to define the concepts of the meta-models that are equivalent with respect to the ontology. For instance, the concept *C* of the source meta-model and the concept *i* of the target meta-model are equivalent as they are respectively mapped with the same concept *3* in the figure. Those equivalent concepts enable to define the consistency of a transformation with respect to the ontology.

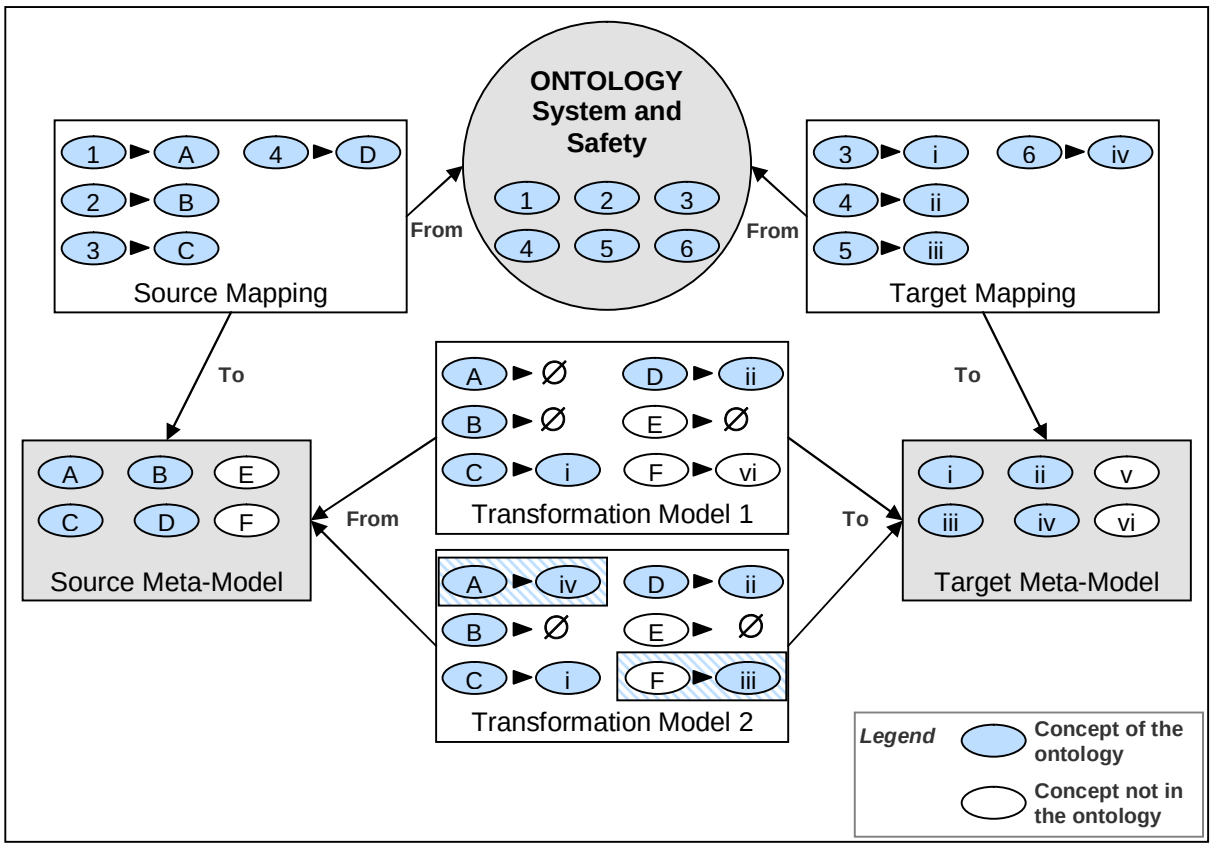

Figure 6. Model transformation framework

In the figure, *Transformation Model 1* is consistent with the ontology as all the transformation rules are consistent with the ontology, i.e., all the source concepts that are mapped with the ontology are transformed into their equivalent target concept and all the target concepts that are mapped with the ontology have been transformed from their equivalent source concept. If a source concept is not subject to transformation (*e.g.*, concept *A* is not the object of transformation) the consistency property still holds. The framework does not allow checking the consistency of concepts outside the ontology so we disregard the transformations of those concepts as they are neither fundamental for systems engineering nor for safety.

Transformation Model 2 illustrates the contrary as source concept *A* is transformed into target concept *iv* and those concepts are not mapped to the same ontology concept (*1* is mapped to *A* and *6* is mapped to *iv*. *1* and *6* are not equivalent). The transformation is inconsistent with respect to the ontology. If a source concept not mapped with an ontology concept is transformed into a target concept mapped with an ontology concept then the transformation is inconsistent (*e.g.*, source concept 6 is transformed into target concept *iii*). Reciprocally, if a source concept mapped with an ontology concept is transformed into a target concept not mapped with an ontology concept, the transformation is inconsistent.

This framework enables to guarantee the interoperability of different tools on the semantic level. The ontology presented in the previous section explains how system and safety engineering have to be understood. Within the framework, we can verify that the SE design tools implement correctly the ontology at the semantic level (*i.e.*, the meta-model implemented by the tool is consistent with the ontology). Moreover, the framework shows how those tools can interoperate seamlessly via model transformations that are consistent with the ontology. Finally, this framework enables to assess if the language and the transformations proposed by a tool correctly implement the ontology and, therefore, if this tool can be used to support the Renault system design process.

Conclusion

In this paper, we presented ongoing initiatives at Renault that aim at introducing formal descriptions in the SE design process. A system and safety ontology has been defined, it integrates the concepts of the ISO 26262 standard, which deals with the functional safety of electric/electronic/programmable systems, with the concepts of the SE design process. This ontology is used as the reference model in a MBSE approach, meaning that all the models used during the design process will have to comply with the ontology. The sensible subject of ontology evolution has been identified but we do not deal with this issue yet. The interested reader can refer to (Haase and Stojanovic 2005 and Ye *et al* 2008) for interesting insights and applications.

Although the initial purpose of these initiatives was to prepare our engineering divisions to the arrival of ISO 26262, we sense that the benefits of using the ontology go beyond the safety aspects alone. We argue that the utilization of an ontology allows the elimination of information losses by improving communication, facilitates the documentation or justification of design choices and improves tool interoperability and component reuse. Finally, the approach presented in this paper ensures the consistency of the whole design process.

Coming back to the subject of system safety, our future work will tackle the ISO 26262 requirements that recommend the use of formal methods as a verification technique. The standard, however, does not give any indication as to which formal method to chose, since this choice depends on the purpose and scope of validation and verification activities. ISO 26262 represents an opportunity to investigate how these methods can help improve the development of automotive systems and how they can be adapted to the specificities of the automotive domain. We have chosen to evaluate the capabilities of the formal languages and formal tools AltaRica, B and Simulink Design Verifier to formally model and verify that the system satisfies safety properties during the different phases of the design process. For these evaluations, the ontology has already helped to identify the similar concepts of the design artifacts and the languages, facilitating the application of the formal methods

References

1. Anneke K., W. Jos and B. Wim. 2003. *MDA Explained: The Model Driven Architecture™: Practice and Promise*, ed. Addison Wesley.
2. Avizienis, CA., J.-C. Laprie, B. Randell, and C. Landwehr. 2004. Basic Concepts and Taxonomy of Dependable and Secure Computing. *IEEE Transactions on Dependable and Secure Computing*, , 1(1): 11–33.
3. Bishop, R. H. 2008. Mechatronic Systems, Sensors, and Actuators : Fundamentals and Modeling. *The Mechatronics Handbook Series, Volume 1 of The Mechatronics Handbook.* 2nd edition.
4. Burr, H., T. Deubel, M. Vielhaber, S. Haasis, and C. Weber. 2005. CAx/Engineering Data Management Integration: Enabler for Methodical Benefits in the Design Process. *Journal of engineering design*, 16(4) :385–398.
5. Chalé Góngora, H. G., O. Taofifenua, and T. Gaudré. 2010. A Process and Data Model for Automotive Safety-Critical Systems Design. In Proceedings of the 20th annual International Symposium of the INCOSE (Chicago, IL). Seattle: INCOSE.
6. Estefan, J.A.. 2008. *Survey of Model-Based Systems Engineering Methodologies (MBSE) - Rev B.* International Council on Systems Engineering. Seattle: INCOSE.
7. Sanford, F., Moore, A. and Steiner, R.. 2008. *A Practical Guide to SysML- The Systems Modeling Language*, Morgan Kaufmann OMG Press.
8. Gruber, T.R.. 1995. Towards Principles for the Design of Ontologies Used for Knowledge Sharing. *International Journal of Human-Computer Studies*, 43 :907–928.
9. International Organization for Standardization (ISO). 2009. Draft International Standard ISO/DIS 26262: Road Vehicles – Functional Safety. Geneva: ISO.
10. Kergosien, Eric, Mouna Kamel, Christian Sallaberry, Marie-Noëlle Bessagnet, Nathalie Aussenac-Gilles and Mauro Gaio. 2010. Construction et enrichissement automatique d'ontologie à partir de ressources externes. CoRR, (abs/1002.0239).
11. Levendovszky, T., G. Karsai, M. Maroti, A. Lede-czi and H. Charaf. 2002. Model Reuse with Metamodel-Based Transformations. In Proceedings of ICSR-7. ed. C. Gacek . Springer Berlin / Heidelberg.LNCS 2319/2002, 166-178.
12. Papadopoulos, Y., Mcdermid, J., Sasse, R. and Heiner, G. 2001. Analysis and synthesis of the behaviour of complex programmable electronic systems in conditions of failure. In *Reliability Engineering & System Safety*, Vol. 71, No. 3. (March 2001), pp. 229-247.
13. Roser, S. and Bauer, B. 2005. Ontology-Based Model Transformation. MoDELS Satellite Events 2005: 355-356
14. Söderström, Eva, Birger Andersson, Paul Johannesson, Erik Perjons and Benkt Wangler. 2001. Towards a Framework for Comparing Process Modelling Languages. In *Lecture Notes In Computer Science*; Vol. 2348. Proceedings of the 14th International Conference on Advanced Information Systems Engineering. 600 – 611.
15. Struss, P. and C. Price. 2004. Model-Based Systems in the Automotive Industry. *AI Mag*, 24(4): 17–34.
16. Sure, Y., Staab, S. and Studer, R. 2002. Methodology for Development and Employment of Ontology Based Knowledge Management Applications. SIGMOD Rec., 31(4) :18–23.
17. Ye, R., Wang, Y., Guo, J. and Xiong, Q. 2008. A Method to Guarantee Ontology

Consistency on Property Range Changes. In Proceedings of IFIP International Conference on Network and Parallel Computing, 2008.

Biography

Hugo Guillermo Chalé Góngora is a specialist in Systems Engineering at Renault. He is currently working as a system architect for systems of systems and is also in charge of the development and the deployment of model-based systems engineering for the vehicle engineering divisions. During the last years, he has been interested in safety-critical systems,, formal methods and architecture description languages and, most recently, in autonomous vehicles. He holds a doctorate degree in thermal and energy sciences (Centrale Lyon, France), a post-graduate diploma on internal combustion engines and environment (IFP School, France) and on energy conversion (ENSAM, France) and a mechanical-electrical engineering degree (UNAM, Mexico).

Thierry Gaudré is a specialist in Systems and Software Engineering at Renault. He is in charge of the development and deployment of specification methods and requirement management tools for systems and software. During the last years, he has worked on Requirements Engineering participated to Systems Engineering training for Renault engineers and technical support for innovative systems projects. In the past, also for Renault, he has worked in the field of Quality assurance and on Dependability of software-intensive systems where he led studies on Verification and Validation techniques by means of statistical tests and formal methods. Thierry Gaudré is a 1992 Engineer graduate from Supélec (France), specialized on Instrumentation and Measurement systems.

Nicole Levy is full professor in computer Science at Cnam, Paris and member of Cedric Laboratory since September 2010. She occupied the same status at University of Versailles Saint-Quentin en Yvelines, being member of the PRiSM Laboratory for 12 years. Prior to that, she has been assistant professor at University of Nancy 1 belonging to the LORIA laboratory. Her research interests include using formal methods to specify complex systems and software architectures. She has led the University of Versailles engineering school, called ISTY, for 5 years. She had a research group at PRiSM, on development and reconfiguration processes for software architectures based on both functional and non-functional properties.

Jean-Louis Boulanger is an Independent Safety Assessor in the railway domain and member of CERTIFER. He obtained his PhD in computer Science in 2006. Prior to that, he worked with many industries of the railway domain. His research interests include requirements, software verification and validation, traceability and RAMS with a special issue for SAFETY.

Ofaina Taofifenua is a doctoral candidate at UVSQ (France) for Renault. His research interests include formal methods and their application to the design of mechatronic systems and the security-innocuousness of safety-critical systems. He possesses a Master in data processing from the University of Bordeaux (France).

Alexandra-Cristina Ţopa is an intern currently working at Renault as part of her Master degree in Complex Industrial System Engineering at “Ecole Polytechnique” (France). She took part in the project "Embedded System Design, tools and methods for Model Transformation". She obtained her bachelor degree from the “Politehnica” University of Bucharest (Romania), in Automatic Control and Computer Science.