An Overview of Software-based Support Tools for ISO 26262

Denis Makartetskiy, Davide Pozza, Riccardo Sisto
Dipartimento di Automatica e Informatica, Politecnico di Torino
e-mail: {denis.makartetskiy, davide.pozza, riccardo.sisto}@polito.it

Abstract

Safety in the automotive domain is becoming more and more important with the ever increasing level of complexity in emerging technologies built-in into the cars. As a stimulus for industry to refine its safety measures related to electrical, electronic and software systems in the cars, the ISO 26262 standard has been recently introduced.
Developing safety-related systems according to this standard in an efficient and effective way requires adequate computer-aided support. For this reason, some initiatives towards software-based supporting tools for ISO 26262 were recently started.
This paper gives an account of the main such initiatives after recalling the main features of the ISO 26262 standard.
In particular, we briefly discuss how the main activities from ISO 26262 such as hazard analysis and risk assessment, functional safety concept, safety validation, etc can have a software support, and what is the state-of-the-art.

1. Introduction

Nowadays, electronic components are pervasive in road vehicles. Indeed, during the last 30 years the majority of innovations on road vehicles (e.g. electronic injection, airbags, ABS, ESP) have been based on electronic embedded systems, and this trend seems confirmed to continue in the future. Therefore, functional safety (i.e. the safety that depends on the system or equipment operating correctly in response to its inputs, including the safe management of likely driver errors, hardware failures and environmental changes) is acquiring more and more importance for the electronic systems that have to be integrated inside cars.
Up to now, automotive companies have developed similar, but custom, safety-related methodologies and processes, and have used pretty much the same set of techniques (e.g. FMEA, FTA) for the development of safety-related electronic embedded systems. However, at the same time, they are not sharing a common view and consideration of the safety of the produced items. In this context, ISO 26262 should become the functional safety reference standard. Although it is not yet published in its final form (it should be in 2011), it represents the shared effort of car makers, OEMs and Tier-1 suppliers to establish a common way to understand and consider the safety concept and its importance, when designing and developing embedded systems for road vehicles. As a consequence, car-makers have already started activities directed to transform their current processes to produce systems that can be ISO 26262 compliant.
One of the problems that emerge when applying this transformation is how to organize the new processes and how to adequately support them by software tools. A first commercial tool that partially responds to this need has recently appeared. Moreover, some collaborative initiatives have emerged in the recent panorama: French car-makers have started in 2007 a project called EDONA (Environnements de Développement Ouverts aux Normes de l'Automobile), while in Italy a project called SiSMA (Sicurezza Funzionale dei Sistemi Meccatronici Automotive) is just starting.
From the technical point of view, an important shared idea about the supporting software tools is that they should provide a means for modelling at the system-level and at different abstraction levels. This kind of models should then be used as a unifying basis on which the safety-related activities are built and synchronized. This modelling facility should be at least semi-formal, in order to respond to the requirements of ISO 26262.
Among the various available possibilities, three semi-formal languages have mainly been considered: plain UML, SysML, and EAST-ADL2 [4]. While the first two are general-purpose solutions, the latter is a domain specific specialization of the former, specifically tailored for modelling automotive embedded systems, that enriches the official AUTOSAR modelling language with additional abstraction levels. AUTOSAR is another important standard that focuses on the design and development of software for automotive embedded systems.
The aim of this paper is to present an overview of the above mentioned current initiatives towards software tools supporting ISO 26262.
In section 2, general information about the ISO 26262 and AUTOSAR standards is outlined. In section 3, the existing projects aiming at bringing integrating solutions for industry to apply ISO 26262 techniques in practice are analyzed. Finally, section 4 concludes.
The definitions that are necessary for understanding the domain will be given where it is needed.

2. ISO 26262 and AUTOSAR

ISO 26262 is an upcoming standard that adapts ISO/IEC 61508 (a standard concerning the safety of systems, that is applicable to all kinds of industry) to the automotive industry. In particular, ISO 26262 addresses the safety related systems that have to be installed in series production passenger cars (with a maximum gross weight up to 3500kg), that are composed of one or more electrical or electronic (E/E) systems.

At the time of writing, ISO 26262 is in the state of a draft international standard (DIS). The figure below shows the simplified structure of its development process (the V-model).

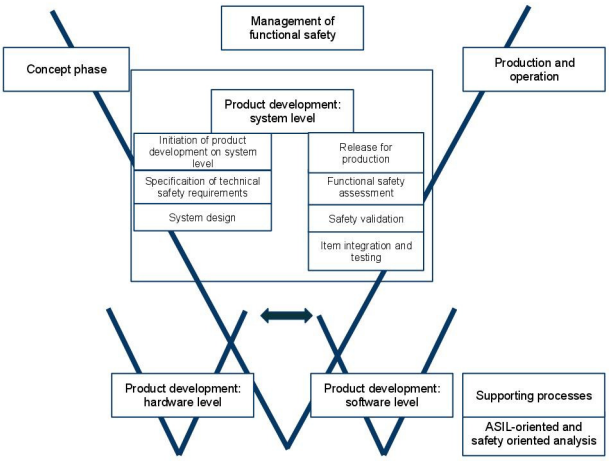

Figure 1. V-Model from ISO 26262

As it is shown in the figure, the V-model development applies to software and hardware development independently on each other and on the overall development process as well.
The standard prescribes the V-model for the development process and describes how functional safety has to be managed during the whole lifecycle of E/E safety-related systems, while providing guidance in the selection of (core and supporting) processes within product development, in function of the outcome of a specific safety analysis methodology. Indeed, the standard is centered around the concept of Automotive Safety Integrity Level (ASIL), that is a qualitative measure of the needed integrity level (i.e. the probability with which a system correctly performs its safety-related functions). The ASIL is determined by means of hazard analysis and risk assessment. According with the standard, since the beginning of the development of systems, each one of its intended functions has to be analyzed with respect to possible hazards. Hence, in function of the probability of exposure to an hazard, the possible controllability by a driver and the severity of a critical event, the risk is estimated and an ASIL is determined. Four ASIL levels have been defined, running from A (lowest) to D (highest). ASILs have to be mapped on the safety requirements that have to be generated to avoid/reduce the identified risks. Thus, it encourages making a focus on safety-critical functions, while not wasting a lot of time on non-critical ones. Therefore, the standard provides requirements for the whole lifecycle and guidance in choosing adequate methods (e.g. hazard analysis, risk assessment, and safety analysis methods) and procedures (e.g. safety, requirements, and document management) to achieve the required ASIL for the developed product.
The standard defines some new terminologies and concepts. It defines the word “item” as a system or array of systems or a function to which ISO 26262 is applied and the process of collecting and describing an item as “item definition”.
The concept phase consists of defining the preliminary architectural and functional design of future system elements and of performing their safety analysis. Safety analysis results in safety requirements that have to be satisfied in order to achieve safety goal (formalized conditions of element's safe functioning). The concept phase is followed by specification of safety requirements into low-level technical requirements and system design. On the hardware and software level the implementation of system functionalities along with technical requirements is done. Then, verification and validation counterparts for safety goal specification and system design take place. Eventually, provided that during system production, safety requirements are not violated, system goes to operation.

Although ISO 26262 derives from ISO 61508, it differs in some valuable points. While ISO 61508 mainly covers industrial equipment and process plant, which are usually produced in small numbers, ISO 26262 focuses on E/E systems for series production cars, hence the standard also covers the requirements for the production of systems in series. Moreover, it is worth noting that ISO 26262 provides much more information and guidance for qualifying and classifying software tools than ISO/IEC 61508. Furthermore, it has to be noticed that controllability was not foreseen by ISO/IEC 61508 to compute Safety Integrity Levels.
There is some connection between ISO 26262 and another important automotive standard called AUTOSAR (AUTomotive Open System Architecture). This is an ongoing motion in the automotive world, started in 2003 and directed to building common open standardized software architecture. For more information about AUTOSAR see, for example, [10], [11], [12].
Since AUTOSAR addresses also safety critical embedded software, it aims at showing relevant compliance with related safety standards (ISO 26262 in this case).

In fact, AUTOSAR defines both a software architecture and a supporting methodology to develop E/E software systems for the automotive domain but cannot guarantee functional safety of such systems by itself. Thus, the implementation of safety-related embedded systems using AUTOSAR has to be done with compliance to related safety standards designed for the automotive domain.
Starting from release 4, the AUTOSAR standard proposes some technical information required proving the ISO 26262 compliance for AUTOSAR members, but it does not provide procedures or activities that address the safety problem by itself.
The full responsibility for implementing the functional safety mechanisms described inside the AUTOSAR framework fully resides on the implementer who will have to fulfill all the specific safety related regulations. This means that AUTOSAR does not include implementation of safety activities into their shared software platform, leaving them for implementation by car makers independently on collaborative AUTOSAR movement, thus competing with each other on this implementation.
The approach to functional safety in AUTOSAR with respect to ISO 26262 concerns mainly the Safety Element out of Context (SEooC) that is a safety element for which an item does not exist at the time of the development. According to ISO 26262, during a system development process, a safety element can be developed as an item (with stricter requirements and more tediously) or as a SeooC (when requirements are substituted with assumptions). For details about SEooC, see [1].
As for modeling, the official AUTOSAR modeling language has been developed as an UML/SysML profile. For details, see [9].

3. Supporting tools and initiatives.

Using software tools can extremely facilitate some of the activities required by ISO 26262 for the development of safety-critical electronic embedded systems. Indeed it would be hard if not impossible to accomplish all the requirements of the standard without an adequate software tool support.
Among the tasks which have to be done in compliance with ISO 26262 and that should be supported by software tools there are: item definition, ASIL determination, ASIL decomposition, hazard analysis and risk assessment activities, safety goal definition and safety requirements allocation, V&V activities safety validation, configuration and change management, etc.
While for most of these activities support software tools already existed before the introduction of ISO 26262, the standard requires a unified management of safety-related activities whereby the various tools need to be integrated as a part of a single framework. This need arises, for example with respect to traceability requirements.
At the best of our knowledge, there are only two main initiatives collecting software solutions for the aforementioned activities under a single roof: the Medini Analyze software tool and the EDONA project and platform. In this chapter we will analyze how exactly they cover these activities and the requirements coming from the standard.

3.1. Medini

Medini Analyze aims at covering all the main ISO 26262 activities during the system development process with a particular focus on safety analysis. It brings together functional architecture design and functional safety analysis, making the main accent on hazard analysis and risk assessment. The structure of the work flow in Medini Analyze reflects corresponding parts of the ISO 26262 V-Model.
Medini Analyze naturally introduces the concept of item into its work flow with the possibility to bind it with any external documentation which can be uploaded into the Medini Analyze work flow.
Within a single project one can manage multiple items, integrating them within the architecture model. Moreover, functions can be added to each item. For example, a cruise control system works by measuring the speed of the vehicle, by estimating the inclination of the driving surface, and by interacting with the vehicle's engine management system. Hence, hazard analysis is made for the chosen single item or for one of its functions. During hazard analysis, the tool guides the user into providing description of the operational situation, the item operation modes, the hazards and the possible malfunctions of the item, so as to obtain a hazard list for the item (or function). For example, for cruise control an hazard could be the unintended acceleration of wheels on a slippery surface without changing the actual speed of the car. Prerequisites, conditions, potential effects of hazardous events can be described.
After the hazard analysis has been completed, the tool provides a wizard for determining ASIL levels. Then, as prescribed by ISO 26262, the tool supports the insertion of a safety goal for each dangerous hazard, along with the inheritance of ASIL levels. A safety goal is a top-level safety requirement, that is defined as a result of hazard analysis and risk assessment and that can be shared by different hazard list entries.).
For example, a safety goal for the cruise control hazard associated with driving on slippery surface can be "do not allow use of cruise control if driving surface is wet". This safety goal inherits the ASIL level of the corresponding hazard.
Afterwards, safety goals can be broken down to functional safety requirements and this is supported with a palette based system to edit SysML structural elements. Figure below shows some possible safety requirements for the cruise control as an example.

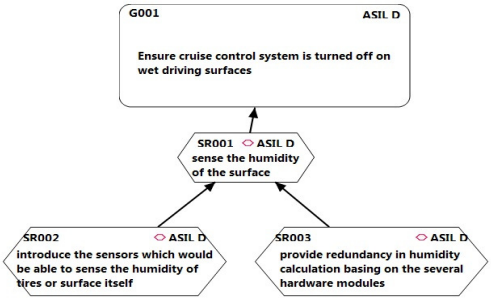

Figure 2. Safety requirements for cruise control by means of Medini Analyze.

The tool embeds support to perform both qualitative and quantitative FTA and FMEA, to facilitate the derivation of requirements. Therefore, it allows the user to create SysML models with the help of an embedded SysML editor. Different levels of abstraction can be created for the defined system, e.g. item level and system level. Architecture SysML models can be used for allocating safety requirements and serves as a base for hazard analysis.
A notable feature of the tool is that it also provides integration with the most popular modeling and development environments in the automotive industry, i.e the Mathworks Matlab/Simulink/Stateflow suite. This is done by linking safety requirements to structural blocks and by the opportunity to see Simulink models inside Medini Analyze.

Traceability
During a system development process it is crucial to have a clear picture of how elements such as requirements, functions, etc are connected to one another and the standard requires these connections to be traced. Because of the complexity of the systems under analysis, shifting the attention from a certain element to a connected one can be not an easy action. Medini Analyze adopts the concept of trace matrix for representing and setting bindings between elements and sub-elements. On the base of this matrix focus is shifted in one-click between connected elements.

Validation
As ISO 26262 sets some rules during activities, there is the need to verify that such rules are fulfilled during the system development process. Let us consider for example *ASIL decomposition*. This is an activity aimed at reducing the likelihood of systematic failures, consisting of substitution of a safety requirement with high ASIL with redundant requirements that have lower ASIL levels. This decomposition can be easily done in a wrong way. To prevent such mistakes a validation engine has been implemented in the tool, so that either the whole project or part of it can undergo the validation to found inconsistencies. The engine makes use of the OCL language, which is a declarative language for describing rules applied to UML models. Therefore, besides the set of rules provided with the tool, users can provide their own rules and/or customize the ones built in the tool.

Document generation.
ISO 26262 requires the production of many documents. Medini Analyze facilitates the production of such documents by generating some of them directly from underlying models and their associated information. For example, "functional safety concept" (the document consisting of safety goals, and safety requirements for these goals) is generated in a completely automatic way by Medini Analyze.

Interfacing and work flow.
Almost all of the file formats used by Medini are XML/XMI, which makes them easy to be included in import/export operations by a wide spectrum of external tools.
For the sake of comfortable user experience, all the activities are made in such a way that they can be used in an iterative manner, without following a predefined sequence of actions and with high degree of independence.
Though Medini Analyze is an Eclipse based platform and extension for other operating systems is planned, at the moment it is available only for the Windows OS.

3.2. EDONA

The EDONA project is a French initiative that aims at constructing an inter-operable integration platform for automotive software development tools allowing the co-development through formalized interfaces over the entire development cycle, rather than providing new tools. The aim is also to integrate safety-based innovations into a common software platform, considering AUTOSAR prescriptions.
The project is directed by Renault, and federates 32 innovating technologies and 13 common source (open-source for members) projects. The project is going to finish in October 2010.
The EDONA components can be generally divided into two parts: Eclipse-based and non Eclipse-based. As a special class of components in EDONA come AUTOSAR components. In Figure 3, the main components (technologies) are shown.

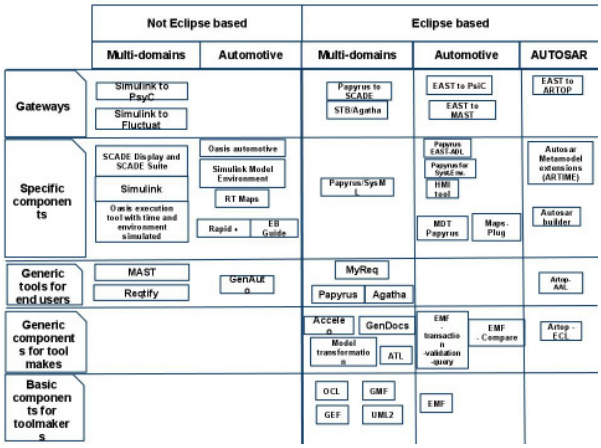

Figure 3. EDONA components.

The framework reuses a set of basic Eclipse components, such as the Eclipse Modeling Framework (EMF) [5] model repository and the ATL Transformation Language (ATL) [6] for model transformation. EMF is a modeling framework and code generation facility for building tools and other applications based on a structured data model, while ATL is a model transformation language and toolkit. In the field of Model-Driven Engineering (MDE), ATL provides ways to produce a set of target models from a set of source models.

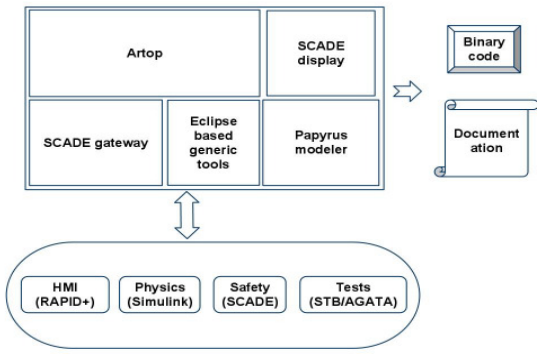

Figure 4. EDONA structure.

The EDONA project aims at integrating tools for the development of embedded software, while addressing dependability, by taking into a particular consideration real-time control systems. This implies requirement and specification traceability, functional and timing validation, deterministic real-time design, as well as HMI (human-machine interface) design and validation with dependability.

For the modelling part, EDONA adopts EAST-ADL and uses Papyrus [13] as the tool for modelling the system.

Safety analysis in EDONA is provided by the Usine Logicielle project. Unfortunately, there is no yet public information about how EDONA covers ISO 26262.

As a means of requirements management, the standalone tool Reqtify [14] and its simplified (as an Eclipse plug-in) version MyReq [14] are used. As an essential activity within the safety critical development process, a large effort has been done towards testing. C code is generated from a Simulink model with the help of the SCADE Suite [8], which is qualified software according to several international safety standards, including ISO/IEC 61508. Then, test cases are generated on the base of source code together with requirements for minimum values of coverage metrics. Test generation is based on Safety Tests Builder IHM [16] and AGATHA [7]. The technique used for test derivation is symbolic automata execution. From Safety Tests Builder IHM, test cases can be passed to a Simulink model for execution or can be exported into an Excel table. For further information about EDONA, see [3].

4. Conclusion

In this paper we have provided an overview of the new upcoming ISO 26262 standard, focusing on the processes and activities that can be supported by tools. Furthermore, we have given a brief overview of Medini Analyze and of the EDONA platform, which are the first initiatives towards integrated software environments for supporting the development of safety-critical electronic components according to the ISO 26262 standard. The aim of these tools is to cover safety-critical design aspects in a richest way in comparison with "one-task" tools, that are already numerous on the market.

Medini and EDONA are just the first steps towards software-based tool support for ISO 26262. It can be expected that more tools will appear on the market in the near future (especially after the official publishing of ISO 26262), and/or that the existing ones will evolve. The target will be to cover the gaps still left in process automation and to get even better integration of the tools composing consecutive tool chains. These targets are being considered by SiSMA, an Italian applied-research project that is just starting.

As it usually happens in software development, a new domain is covered by commercial software firstly, and then, with some delay, open-source solutions appear. Accordingly, new open-source software initiatives giving specific ISO 26262 support can be expected to rise later in the future.

Along with tools, a fast development of safety aspects' support in standardization of automotive system development, which has began to evolve with the publication of AUTOSAR release 4.0, is also expected.

Acknowledgments

We are grateful to the Management Team of ikv+ for a given opportunity to evaluate Medini Analyze with a trial version.

This paper has been written thanks to the support of the SiSMA project.

References

1. ISO 26262 “Functional safety of road vehicles”, 2009.
2. Born M., Favaro J., Kath O. “Application of ISO DIS 26262 in Practice” *Proceedings of the 1st Workshop on Critical Automotive applications: Robustness & Safety*, ACM, 2010, pp.3-6.
3. Ougier F., Terrier F. “Eclipse based architecture of the EDONA platform for automotive system development” *Proceedings of ERTS'2010*.
4. Cuenot P. et al. “Towards Improving Dependability of Automotive Systems by Using the EAST-ADL Architecture Description Language”. *Architecting dependable systems 4. Section: Architectural description language*, 2007, pp.39-65.
5. Budinsky F., Brodsky S., Merks E “Eclipse Modeling Framework” Pearson Education. ISBN:0131425420, 2003
6. Jouault F., Kurtev I “Transforming models with ATL” *Satellite Events at the MoDELS 2005* Conference, Springer Berlin, pp.128-138
7. Bigot C., Faivre A., Gaston C., Simon J. “Automatic Test Generation on a (U)SIM Smart Card” in *Smart Card Research and Advanced Applications*, pp.345-358, 2006
8. Abdulla P., Deneux J.,Stalmarck G., Agren H. and Akerlund O. “Designing Safe, Reliable Systems Using Scade” in *Leveraging Applications of Formal Methods*, Springer Berlin/Heidelberg, pp.115-129, 2006
9. AUTOSAR document. Release 4.0. “UML profile for AUTOSAR”, 2010.
10. AUTOSAR document. Release 4.0. “AUTOSAR Technical Safety Concept”, ID 362, 2010.
11. AUTOSAR document. Release 4.0. Requirements on Methodology, 2010.
12. www.autosar.org
13. www.papyrusuml.org
14. www.geensoft.com/en/article/reqtify/
15. http://sourceforge.net/projects/myreq/
16. http://www.chiastek.com/products/stb.html