Validation & Certification of Safety-Critical Embedded Systems - the DECOS Test Bench

Erwin Schoitsch[1], Egbert Althammer[1], Henrik Eriksson[2], Jonny Vinter[2], Laszlo Gönczy[3], Andras Pataricza[3], and György Csertan[3]

[1] (ARC Seibersdorf research, Austria)
{erwin.schoitsch, egbert.althammer}@arcs.ac.at
[2] (SP Swedish National Testing and Research Institute)
{henrik.eriksson, jonny.vinter}@sp.se
[3] (Budapest University of Technology and Economics)
{gonczy, pataric, csertan}@mit.bme.hu

Abstract. The integrated EU-project DECOS (Dependable Embedded Components and Systems) aims at developing an integrated architecture for embedded systems to reduce life-cycle costs and to increase dependability of embedded applications. To facilitate the certification process of DECOS-based applications, the DECOS Test Bench constitutes a framework to support Validation & Verification. By implementing a modular approach, an application safety case merely contains the application-specific issues and re-uses the safety arguments of the “generic” safety cases of the DECOS platform. The Test Bench covers the complete life cycle from the platform-independent models to deployment, including model validation and transformations. The safety cases are based on validation-plans (v-plans) comprising the steps to validate the safety requirements. The Test Bench provides a methods/tools repository, guidelines to generate and execute v-plans, and integration of tools and of remotely distributed test beds.

1 Introduction

“Smart Systems” are based on intelligent embedded control systems, which are distributed within the application systems, and often hidden to the every-day life user. E.g., more and more functions in today’s cars are realized by electronics and software, 80-90% of the new innovative features are realized by distributed embedded systems. Eventually, even highly safety critical mechanical and hydraulic control systems will be replaced by electronic components. Value of electronics in cars will increase beyond 40% of the total value. Even today, upper class cars contain up to 80 ECUs, several bus systems, and about 55% of all failures are caused by electronics, software, cables and connectors [3], [11].

The DECOS project [2] aims at making a significant contribution to the safety of dependable embedded systems by facilitating the systematic design and deployment

[1]Research supported in part by EU IST-FP6-511764 (DECOS)

of integrated systems [1]. DECOS (Dependable Embedded Components and Systems) is a European Integrated Project in FP6, Embedded Systems area, scheduled for the period 2004-2007. Co-ordinator is Austrian Research Centers Seibersdorf research (ARCS). Work is performed by 19 partners: Two research centers (ARCS, SP (Sweden)), six universities (Universities of Technology Vienna, Darmstadt, Budapest, Hamburg-Harburg, Universities of Kassel and Kiel), three technology and tool providers (TTTech Vienna, Esterel Toulouse, Infineon Munich), and eight demonstrator/application related industrial partners (Audi AEV, CR Fiat, Hella, Airbus, Thales, EADS, Liebherr Aerospace, Profactor).

In federated systems, each application subsystem is located on a dedicated processor. The federated approach provides natural separation of application functions, but causes increased weight, electric energy consumption and cost due to resource duplication and the large number of wires, buses and connectors. Integrated systems not only help to alleviate this problem, they also permit communication among application functions. A remarkable feature of the integrated DECOS architecture is that hardware nodes are capable of executing several tasks of application subsystems of different criticality. Throughout this paper, we will use the notion of a node instead of processor or component.

An integrated architecture provides a fixed number of nodes, each of which has certain properties (e.g., size of memory, computational power, I/O resources). All tasks have to be allocated such that given functional and dependability constraints are satisfied. This is discussed in detail in [4].

This paper focuses on the description of the DECOS Test Bench which supports the validation and certification process within DECOS. It has been developed within subproject 4 (validation and certification). An overview of the other subprojects is found in [5].

2 Goals of the DECOS Test Bench – Modular Certifiability

The DECOS Test Bench has the goal to facilitate *certification* of DECOS-based systems in a *modular* (component based) manner, making use of properties of the DECOS core services, high level services, and the DECOS design and development processes. Basis is the generic functional safety standard IEC 61508. A comparison and evaluation of several domain specific standards and IEC 61508 has shown, that systems conforming to higher SIL levels of IEC 61508 or related standards fulfill the major requirements of domain specific standards, such as IEC 50129 (railways), the evolving ISO 26262 automotive functional safety standard (the so-called DIN-FAKRA standard proposal) or RTCA/DO 178B for aircraft industry.

The modular certification of a DECOS-based system is based on the definition of so-called *Modular Safety Cases* and works in the following way:

- The first step is to provide *generic safety cases* for the *application independent parts of DECOS*, e.g. the core-services or the DECOS nodes. For each entity a safety case is generated. These safety cases are called generic safety cases because they provide the generic infrastructure for any DECOS application but are (per definition) independent from them (comparison: generic telephone infrastructure and applications). The generic safety cases themselves form a modular system.

- The second step is to provide the safety cases for each *application* of the system, re-using the generic safety case(s) for the DECOS infrastructure technology.

Fig. 1 shows the two approaches to establish a safety case. On the left hand side the traditional (federal) approach is shown: one large monolithic safety case for each application. On the right hand side the modular approach is illustrated: reuse of the architecture related (generic) safety cases such as the DECOS nodes, the core services and the high level services. This approach leads to a well manageable certification process and to a clearly arranged application safety case.

The modular, component-based approach to safety case generation is of utmost importance in case of re-certification of DECOS-based systems: based on the generic safety cases and the already proven properties and evidences, re-certification is considerably simplified, the process more cost-effective.

Within DECOS only certifiability will be proven: it is up to the subprojects to prove that the requirements have been met or can be met in a production process (“certifiability”). Doing so, the certifiability of the whole DECOS system is proven.

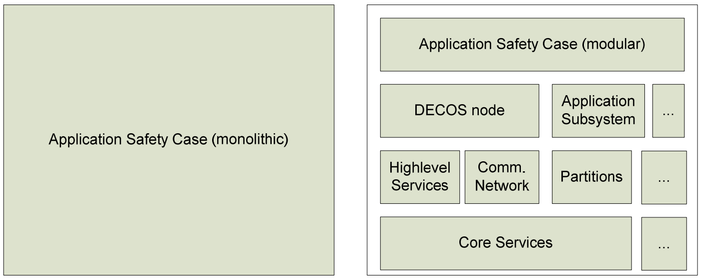

Fig. 1. Monolithic versus modular application safety case

At this stage, one “Generic Safety Case” for a safety critical part of a DECOS node has been established. It builds on the inherent assumptions and the assumed fulfillment of the requirements defined in all the subprojects for the integrated DECOS architecture and services (architecture claims, core services, high level services) [1].

The construct to be looked at is shown in Fig. 2. It consists of the safety-critical part of a DECOS node, including the SCCU (safety–critical connector unit) and the PI (Platform Interface) (for details of DECOS nodes, see [1]), but not of the application, because only the generic part is looked at. The result is an evaluation and assessment of the contribution of the DECOS architecture to the safety of application systems, which is intended to be included in the safety case of DECOS-based applications and is expected to facilitate this part of the system certification.

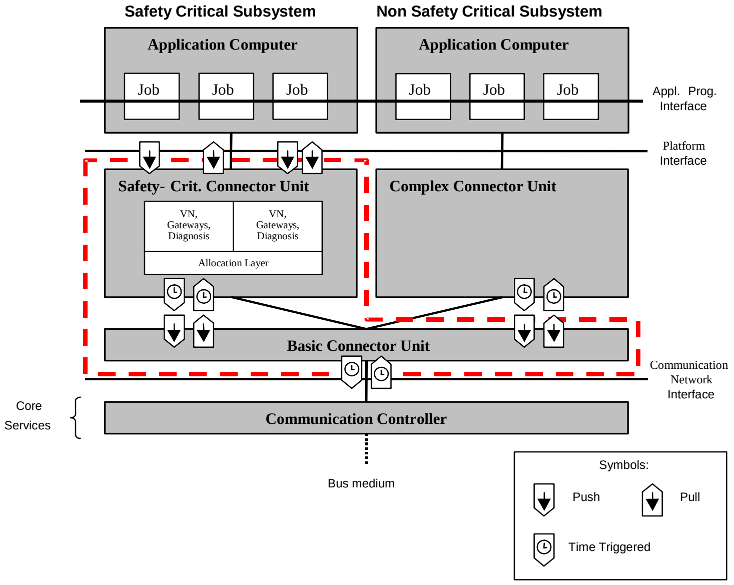

Fig. 2. DECOS node, system boundary for generic safety case

3 DECOS Test Bench Structure

The primary outputs of the DECOS Test Bench are the necessary documents to prove the certifiability of a DECOS-based system, i.e. the required (generic) safety cases. The Test Bench shall therefore allow for generating, organizing, storing and exporting the safety cases. What does this mean precisely? Typically a safety case comprises the necessary safety arguments which correspond to the *V&V activities* and the related *evidence* (for details see section 4). A V&V activity is related to one or more safety requirements and is a necessary step to prove these requirements. A V&V activity is performed either manually or by means of a V&V tool according to the selected V&V method. The evidence is the (written) proof that the V&V activity has been completed with positive results. Note that there is no evidence in case the V&V activity has failed (negative results) or been indecisive (inappropriate V&V tool).

In order to further simplify the certification process, the Test Bench provides an initial list of requirements and related V&V activities for each artefact under test (AUT, e.g. ranging from DECOS metamodels to architecture models, DECOS tools, components (hardware and software, see fig. 8), DECOS applications or DECOS application (sub) systems). The proposed V&V activities in this list are also influenced by the chosen standard and the required safety integrity level (according to IEC 61508 or related standards). The list of the V&V activities for a certain AUT is called a *validation plan* (v-plan), thus the initial list is called initial v-plan.

The DECOS Test Bench provides a number of facilities which provide guidance to generate and execute v-plans to obtain the evidence. The v-plans are explained in section 4. Among the facilities provided by the Test Bench are predefined v-plan templates and the possibility to integrate V&V tools.

Conceptually, the DECOS Test Bench constitutes a *framework* for the consolidated collective *V&V capabilities* corresponding to the DECOS artefact categories, providing v-plans to control the respective V&V activities in a progressive integrated manner, as indicated in Fig. 3.

Depending on the AUT, an appropriate initial v-plan is established, considering all related requirements, and describing the start-up activities for the corresponding V&V process. During progress of this process, this list is continuously updated by either marking individual activities as completed, splitting activities into sub-activities etc. Each activity is linked to one *V&V method* and/or *tool* which is to be used to perform the respective V&V activity. Typical V&V methods for various technologies and life cycle phases (notated in brackets) are

- FTA, FMECA and Hazop (system analysis and evaluation)
- Theorem proving and model checking (formal methods)
- Application of UML and MatLab/Simulink (simulation and modelling)
- Audit, inspection (review)
- Functional, white box and black box testing, coverage, static analysis (testing)
- SWIFI and EMFI (fault injection)
- Conducted and radiated emission (EMI)

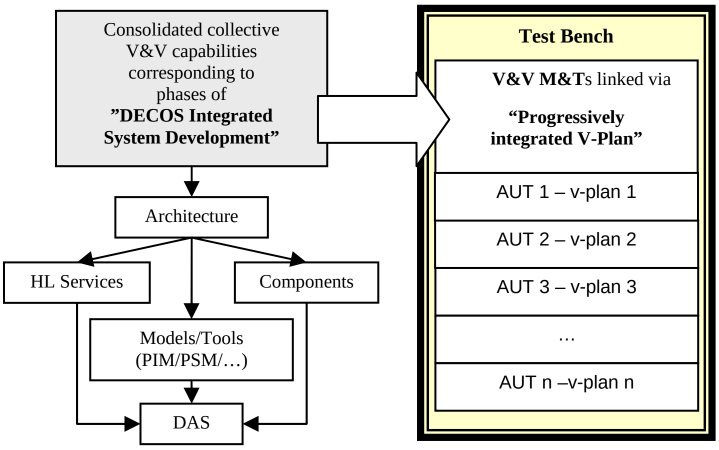

Fig. 3. Relationship between DECOS artefacts and Test Bench structure

A v-plan therefore is always specific for one AUT, comprises the list of requirements to be fulfilled for that AUT, describes the set of associated V&V activities, and contains meta-information like responsibilities and details about the AUT.

Safety cases are related to v-plans since the establishment of a safety case requires the collection of safety evidence for the claimed dependability of the respective system (or artefact), and v-plans describe how these arguments could be verified and validated.

An important aspect is that the Test Bench shall support treatment of more than one AUT at a time, illustrated in the parallelism of the various v-plans.

A top-level view of the Test Bench framework is shown in Fig. 4. Vertical alignments indicate 'uses'- or 'consists of'-relationships. So, a safety case uses a v-plan and the evidence based on positive results of V&V activity, while a v-plan essentially consists of the requirements (or claims, respectively) to be satisfied for a certain AUT, and the V&V activities performed in order to satisfy the requirements. A V&V activity is either a test case generation, or the application of a V&V method. The latter is performed by means of a certain V&V tool assigned to it.

Horizontally, major information flows are indicated. The whole process is requirements driven - requirements are derived from the DECOS artefact, relevant standards, and potentially other sources, e.g. – depending on the AUT category – from the respective domain. For instance, for an automotive brake-by-wire application, SIL3 would be a domain requirement, while for an aerospace flap-control SIL4 or equivalent would be required. In addition, requirements may also be generated internally during the V&V process, e.g. from risk analysis or by substituting general requirements, which cannot be verified as such, by more specific ones. V&V tools use the AUT in its appropriate form (specification, model, software, hardware etc.) – also called 'incarnation' – and produce results. Positive results are used to establish evidence for the validity of the stated requirements, while negative results will be reported to the developer. (Of course, also positive results will be reported to developers.) After error correction, new test cases may have to be added, and failed activities repeated. Test cases are considered to be requirements, since they have to be treated in essentially the same way; however, in addition to be entered manually, they can also be generated by means of test case generators. Finally, all collected evidence establishes the arguments for certifiability of the AUT.

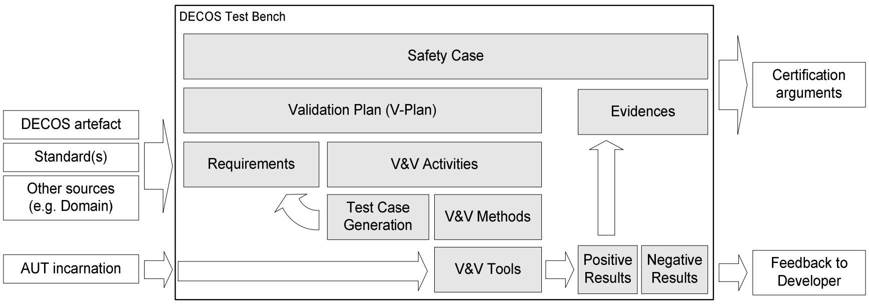

Fig. 4. Top-level structure of the DECOS Test Bench framework

Therefore, the DECOS Test Bench consists of the consolidated set of V&V tools, and a set of software services for guided use of these tools as well as management of all related documentation. These services establish the Test Bench framework, and largely coincide with the ingredients listed illustrated in Fig. 4.

In the following, specific aspects of the Test Bench framework are described: these are v-plans and evidence (section 4), execution of V&V activities (section 5), integration of external tools using model transformation (section 6) and the EMI part of the Test Bench (section 7).

4 V-Plans and Evidence

As described earlier, the v-plan describes the sequence of verification and validation activities. Within the DECOS project an example v-plan has been established for validation of the FTCOM layer. The FTCOM layer is a middleware layer of the DECOS platform which provides fault-tolerance (FT) and hardware-accelerated communication (COM) services to the application. The FTCOM layer is especially suitable as an example since it both consists of software and hardware. The hardware part is challenging due to the fact that it is an FPGA which functionality is designed using a hardware description language, i.e. software.

The DECOS platform shall be able to accommodate applications of highest safety integrity and since DECOS uses IEC 61508 as base standard, the FTCOM validation plan is consequently based on the requirements put by IEC 61508 on SIL4 safety functions. In its present form, the v-plan does not cover the complete safety life cycle of IEC 61508, instead it has been focused on the phases that have a correspondence in a development model such as the v model, see Fig. 5. In other words, concept and maintenance phases have e.g. been left out.

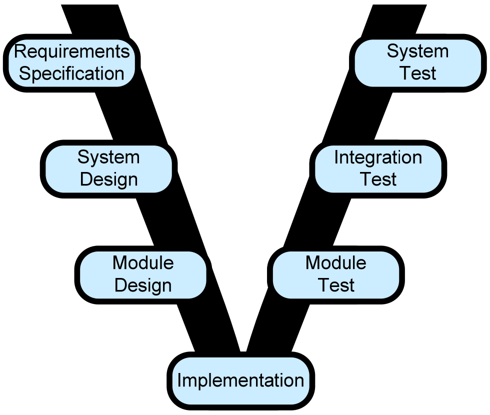

Fig. 5. Modified V-model from the IEC 61508 standard

In order to find all necessary V&V activities which will constitute the FTCOM v-plan, all methods that are highly recommended (HR), recommended (R), or mandatory for SIL4 safety functions to avoid systematic failures during the different phases of the safety lifecycle have been gathered; both for the hardware (IEC 61508-2) and the software (IEC 61508-3) parts. Additional requirements on e.g. document, configuration, and version management as well as requirement formulation have been captured in checklists.

Even though the FTCOM layer only provides 11 services, 4 hardware based and 7 software based, the vast number of V&V activities imposed by IEC 61508 makes it practically impossible (due to limited resources) to carry them all out within a three year's research development project such as DECOS, which is clearly pre-competitive and not developing a product. Instead, a few carefully selected V&V activities will be carried out as a proof-of-concept and to demonstrate certifiability. This is neither a drawback of DECOS nor of the processes implemented, and their feasibility is demonstrated by the application demonstrators in the automotive, aerospace and industrial control area. Guidelines will be provided how the processes have to be carried out in case of product certification and attention is directed to the critical issues. The selection of activities is based on the competences of the DECOS partners and the availability of a specific V&V tool. The V&V activities performed within the DECOS project range from reviews, static analyses, and formal verification to testing and fault injection. Different V&V activities apply for different DECOS artifacts. (see section 3 and 8).

5 Execution of V&V activities

This section looks at one detail of the v-plan execution: the execution of the V&V activities. For simplicity reasons we consider the execution of only a single V&V activity in this context. Note that the Test Bench allows for parallel execution of several V&V activities due to its distributed client/server architecture[2]. As shown in Fig. 6, each V&V activity in the Test Bench has a well defined life cycle (note that the interaction with the requirements is explained below). It starts with the state "Not Ready" which means that the V&V activity has been defined but is lacking the relevant input which is necessary for execution. If the necessary input has been provided and the V&V tool has been selected, the V&V activity passes into the next state which is called "Ready". The input for the V&V activity comprises the input data requested by the tool to produce a significant output, further the definition of a deadline, and a responsible person. The Test Bench provides guidance by offering a repository of V&V methods/tools and an adequate help file. It also supports the input preparation for the selected V&V tool by offering model transformation (for details see section 6).

At this point the Test Bench offers the possibility to implicitly change to the state "Processing": an e-mail is generated which endows the responsible person with the relevant information he or she needs to execute the V&V activity. The responsible person is supposed to directly work on the Test Bench (by having installed a client on its work station). Depending on the selected V&V tool the tool execution might be automated using e. g. a message server (for details and an example see section 6) so

[2] Based on Telelogic DOORS®

that the results are automatically fed back into the Test Bench. In case of a manual execution of the V&V activity the Test Bench provides user guidance by offering a dialog which leads the user step-by-step through the process. An manual execution is typically required for GUI based tools or in case of distributed test sites needing specific equipment (e.g. EMI – Electromagnetic Interference – tests through a test lab, see section 7). The workflow support for the validation and certification process is designed in a manner that the Test Bench can be distributed and include remote sites on both levels of integration. The Test Bench provides a distributed service for many clients and distributed test sites, based on the workflow described and a common repository for reuse of V&V results and documents.

Depending on the results of the V&V activity, there are three possibilities of the consecutive states: "Completed" if all results are positive, "Failed" if not all results are positive, in other words that there is at least one negative result and "Indecisive" if the selected V&V tool turns out to be not appropriate to cover all required aspects. In the second and third case the V&V activity shall be repeated using an updated AUT or V&V tool, respectively, setting it back to "Not Ready". In case of a new version of the AUT the V&V activities of the associated v-plans have to be repeated as well (regression test).

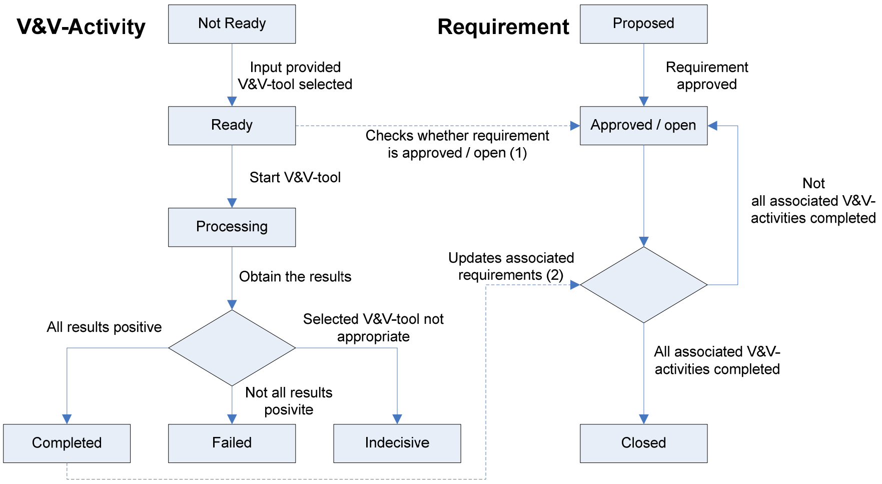

Fig. 6. Life cycle of a V&V activity and an associated requirement

Apart from this life cycle process of the V&V activities the Test Bench supports and/or automates further processes.

It automates the interaction with the associated requirements (see dashed lines in Fig. 6): (1) in the state "Ready" the associated requirements are checked whether they are already set to "Approved / open" which means that the requirement has been approved and is stable (note that the initial state "Proposed" denotes the requirements which are not approved yet). (2) In the state "Completed" the associated requirements are checked whether they have already been fully validated (i.e. all the associated V&V activities are completed). In the positive case they are set to "Closed".

The Test Bench supports a tree-like hierarchy of V&V activities: V&V activities shall be split into sub V&V activities in case more than one V&V tools are used to perform the required task. Such V&V activities are called “compound” then, in contrast to “elementary”. Also a v-plan can be seen as a compound V&V activity. The state of a compound V&V activity is automatically set to the logically lowest state of its sub V&V activities (according to Fig. 6, the order from lowest to highest corresponds to the direction from top to bottom and from left to right in the final row).

The Test Bench supports modular certifiability by allowing reference from a V&V activity to existing safety cases/arguments. It allows the reuse of modular safety cases (or successful V&V activities). Thus, the Test Bench provides a repository (or a library) of reusable safety arguments.

The Test Bench supports the generation of documents in standard format out of the database information. This allows for automating the generation of safety cases. The safety case contains the relevant evidence, but to keep the safety case slim it shall contain only the reference to the evidence which is downloaded from the document repository on request. The overall Test Bench Process as described is shown in Fig. 7.

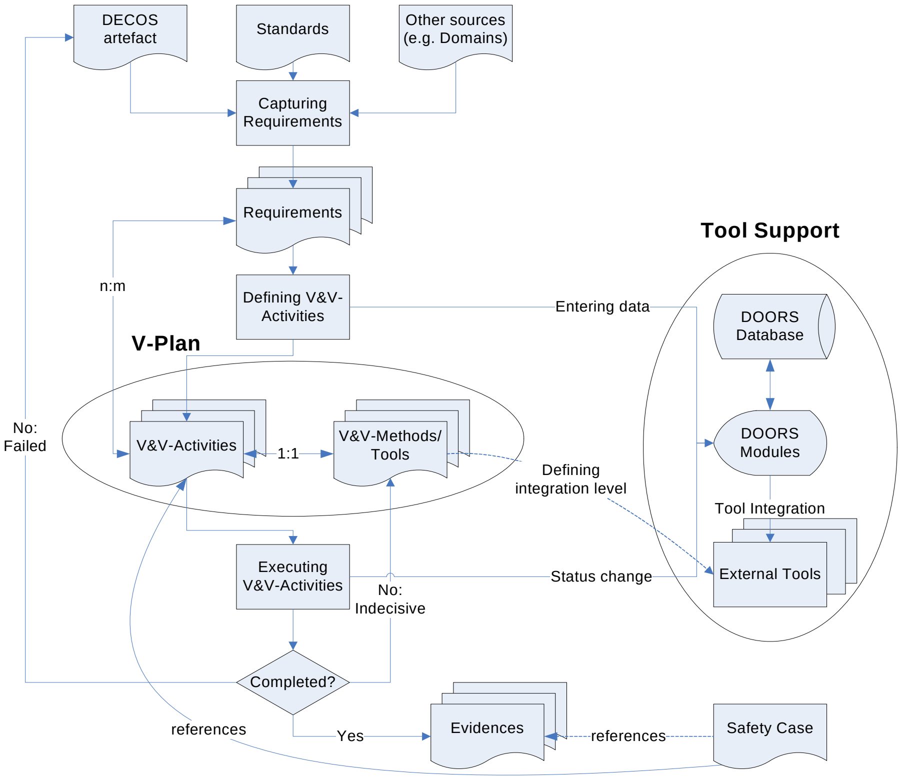

Fig. 7. The overall validation & certification process of the Test Bench

6 Integration of External Tools Using Model Transformation

One goal of the Test Bench is to provide guidance for the integration of external tools in order to facilitate the execution of the V&V activities. One part of the integration is the preparation of the input data for the V&V tools. This is done using model transformation. Hereby we describe the test bench configuration by the example of the PIM (Platform Independent Model) validation as a case study to illustrate the functioning of the Test Bench. In this step a PIM is validated against the DECOS PIM metamodel using ontology. The development process of a DECOS application is shown in Fig. 8, a part of which is the PIM generation.

In the case study, the PIM is edited by a UML modeling tool and stored in XML. Therefore, it has to be transformed into the representation format of an ontology tool and then it can be validated by posing questions against a reasoning system. The first prototype of this transformation was written in XML Schema Transformation Language [6], but a new version is also implemented in the VIATRA [7] tool by using model transformation.

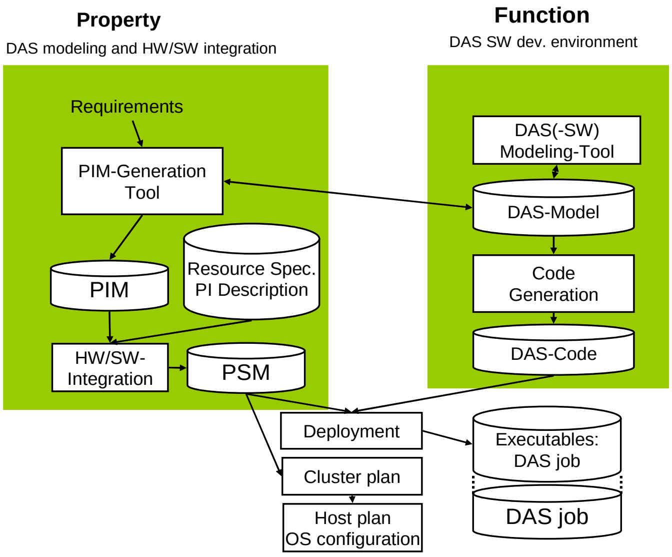

Fig. 8. DECOS development from model to deployment

As multiple users may submit their AUTs for a particular test to the same V&V tool, the DECOS test bench architecture should support the message-driven test evaluation in order to run the memory and CPU time consuming V&V tools in multiple instances on the same server. The well-proven Message Queuing (MQ) technology is proposed as a messaging middleware. Hereby we discuss the concrete implementation considerations, and present the architecture to run the test prototypes on.

The configuration of the implementation is shown in Fig. 9.

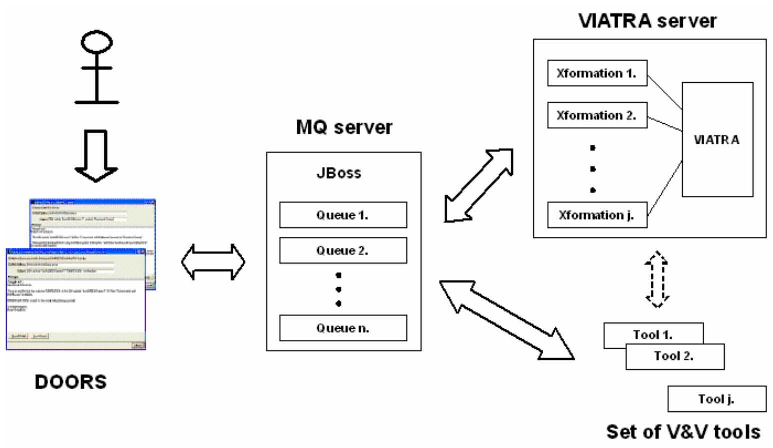

Fig. 9. Configuration of the Test Bench

The core of the basic architecture is a Java Message Service ([9]) compliant messaging server. The consumers and the receivers are the wrappers around the test bench tools sending test requests and results. We have chosen to use the JMS implementation of the JBoss [10] as it is a widespread, open product. The central messaging server and the test bench tools, such as the VIATRA framework and RACER are running on separate machines, thus the architecture is more flexible and extendable.

The requirements are stored in the DOORS tool. Sending a PIM to the test bench needs only a single Java wrapper class which takes the model, or a reference to the model stored in a repository, and sends it to a queue on the server. A concrete queue represents one test tool; in this case, VIATRA2. As the DOORS tool is able to start an executable file, this class can be instantiated from a DOORS script. Messages are extended with a unique message id to correlate response messages.

7 The EMI Part of the Test Bench

In a distributed safety-critical embedded system communication is vital. To be able to design and analyse different cable types and network topologies for the interconnection bus, a dedicated EMI test bed is developed within DECOS. The EMI test bed consists of two parts: one based on software (simulation program) and one based on hardware (measurement set-up). When designing these test beds, relevant standards from all three application areas, automotive, aerospace, and industrial control, have to be considered (e.g. MIL-Std. 461-E, DO 160D, EMC Directive 89/336/EC) . In the EMI test bed it is possible to analyse emission and susceptibility both for narrow band and wide band disturbances.

The EMI hardware test bed is set up inside an anechoic chamber of the accredited EMC test lab at ARCS where the cable under test is mounted in an adjustable fixture and driven and loaded by appropriate circuits, nodes. Three different antennas are provided to cover the targeted frequency range from 150 kHz to 1 GHz.

The EMI software test bed, or simulator, is an enhancement of an existing in-house developed tool at SP Swedish National Testing and Research Institute. The topology of the simulated bus as well as the resulting electric field distribution is graphically presented. Experiments carried out using the EMI hardware test bed will be used to fine tune the EMI software test bed.

8 Conclusions and Future Work

The DECOS Test Bench supports the modular certifiability of integrated dependable systems (such as the DECOS-based systems) according to the safety standards, e.g. IEC 61508, which is a central goal of the DECOS project. For this purpose it provides the Test Bench framework (based on Telelogic DOORS®) which manages the requirements, v-plans and V&V tools and provides guidance to execute v-plans and to obtain the evidence. Since each v-plan is associated to one artifact of the DECOS system, it is straightforward to establish a generic safety case for that artifact. An application safety case thus only contains the application-specific evidence and reuses the results of the generic safety cases.

A further strength of the DECOS Test Bench is that it supports the integration with external tools. In many cases the available input data does not fully match with the required formats of the tool. For this reason it provides a generic way of model transformation based on the tool VIATRA2. This is also true for external test sites such as the EMI Test Bed.

In the automotive application, a mixed criticality approach is demonstrated by integrating a door-control system and a critical crash warning and avoidance system demonstrator (vehicle and environment simulator with DECOS hardware in-the-loop, based on Layered FlexRay core technology).

The aerospace demonstrator is a flap control system for the Airbus outer flap control, a really critical application, with a gateway to the AFDX-bus of Airbus.

The industrial control demonstrator is control of a production- and business critical vibration control system for high-end nano-imprinting machines, controlling piezo-electric sensor and actuator networks. The long term vision of this demonstrator is critical structural control of engineering structures (helicopter cabins, aircraft wings, buildings, noise suppression etc.).

The next steps will be the following:

- Identification of a basic set of (existing) V&V tools to be integrated into the Test Bench to fulfill the major V&V requirements of the project (e.g. Item from Itemsoft for FMECA, LDRA (static and dynamic testing, coverage, top level test case generation), SCADE-MTC, a SWIFI (Software Fault Injection) tool like Propane from TU Darmstadt, a tool for PIM Validation (already explained in section 6) with the implementation of the appropriate model transformations
- Establishment of further v-plans and evidence for the generic safety cases

- Experiments (e.g. on quad modular redundancy by EMFI (EMI fault injection), evaluation of other basic core architectures such as Layered FlexRay and TT-Ethernet (time-triggered), evaluation of the impact of SoC (System-of-Chip) implementation of a few DECOS high level services, and a vulnerability analysis/trust case experiment) which are accomplished in the realm of DECOS.

9 References

[1] H. Kopetz, R. Obermaisser, P. Peti and N. Suri, “From a Federated to an Integrated Architecture for Dependable Embedded Real-Time Systems”, Vienna University of Technology, Austria, and Darmstadt University of Technology, Germany, 2004

[2] DECOS: Dependable Embedded Components and Systems, Integrated Project within the EU Framework Programme 6, http://www.decos.at

[3] Association of German Car Manufacturers (VDA). HAWK2015 – Challenges for the automotive supply chain. Henrich Druck + Medien GmbH, Schwanheimer Strasse 110, D-60528 Frankfurt am Main, 2003 (in German).

[4] G. Weißenbacher, W. Herzner, E. Althammer, “Allocation of Dependable Software Modules under Consideration of Replicas“, Proceedings of the ERCIM/DECOS Workshop on Dependable Software-Intensive Embedded Systems at Euromicro 2005, Aug. 31-Sept.1, 2005, Porto, Portugal. Proceedings published by ERCIM (European Research Consortium for Mathematics and Informatics), ISBN 2-912335-18-8, p. 51-58.

[5] E. Schoitsch, “The Integrated Project DECOS, From a Federated to an Integrated Architecture for Dependable Safety-Critical Embedded Systems – an Overview”, Proceedings of the ERCIM/DECOS Workshop on Dependable Software-Intensive Embedded Systems at Euromicro 2005, Aug. 31-Sept.1, 2005, Porto, Portugal, Proceedings published by ERCIM (European Research Consortium for Mathematics and Informatics), ISBN 2-912335-18-8, p. 9-14. www.ercim.org

[6] XSL Transformations (XSLT) Version 1.0 W3C Recommendation 16 November 1999 http://www.w3.org/TR/xslt

[7] The VIATRA2 Model Transformation Framework, Generative Model Transformer Project, The Eclipse Foundation. http://eclipse.org/gmt/

[8] Volker Haarslev, Ralf Möller, Michael Wessel: RACER User’s Guide and Reference Manual Version 1.7.19

[9] Java Message Service Spec. Version 1.1 http://java.sun.com/products/jms/docs.html

[10] JBoss Application Server. JBoss Inc.. http://labs.jboss.com/portal/jbossas/index.html

[11] E. Schoitsch, "Design for Safety AND Security of Complex Embedded Systems: A Unified Approach"; invited presentation des NATO Advanced Research Workshops, TU Gdansk, Springer, “Cyberspace Security and Defense: Research Issues”, p. 161-174, ISBN-10 1-4020-3380-X, Springer Dordrecht, Berlin, Heidelberg, New York.