Can Cars Fly?
From Avionics to Automotive:
Comparability of Domain Specific Safety Standards

Matthias Gerlach
OpenSynergy GmbH
matthias.gerlach@opensynergy.com

Stephan Weißleder, Robert Hilbrich
Fraunhofer FIRST
{stephan.weissleder|robert.hilbrich}
@first.fraunhofer.de

Abstract

The integration of different functions and infotainment in the automotive domain is a well-known issue. Integration and communication of different safety-relevant and non-safety-relevant functions can be done using microkernel-based operating systems. For the use in avionics, such systems have been developed according to the avionics safety standard DO-178B. For automotive, safety-relevant functions have to meet the requirements of ISO 26262.

This paper presents ongoing work on establishing comparability of safety standards from the automotive domain and the avionics domain. A high-level comparison and mapping of processes and work products between ISO 26262 and DO-178B is provided. In addition, a representative use case as a basis for a case study is presented.

1 Introduction

Current topics in the automotive domain include safety by virtue of ISO 26262 [4], a common automotive E/E architecture (AUTOSAR) [1], and integrated infotainment solutions. For the safe and secure coexistence of different pieces of software with varying levels of criticality on a single hardware platform, microkernel-based operating systems are discussed. A similar approach exists in the avionics domain: Integrated Modular Avionics (IMA). Microkernels as part of IMA have already been developed and certified according to DO-178B [6] for avionics.

ISO 26262 is a derivative of the safety norm IEC 61508 [3] and planned to be published in 2011. By contrast, DO-178B has been published as early as 1992, and is well-established in the avionics domain. A revised version, DO-178C, is expected to be available in 2011. Based on the question of how a DO-178B certifiable microkernel helps development of an ISO 26262 conformant item, this paper presents findings from a comparison of ISO 26262 with DO-178B as a foundation to establish comparability of the two domain's standards. Note that hardware and systems related aspects are specified in separate standards for the avionics industry.

The paper is structured as follows. Section 2 provides a comparison and a proposed mapping between confirmation measures, processes, and work products between the two standards. In Section 3, a case study is outlined, based on commercially available products COQOS [5] and PikeOS [7]. PikeOS represents a microkernel certifiable to DO-178B level B and COQOS enables integrated automotive applications by using AUTOSAR. Section 4 provides a summary and discussion of the findings in this work.

2 Comparing ISO 26262 and DO-178B

For software development, both standards DO-178B and ISO 26262 focus on integrated safety measures. In contrast to external safety measures, they define processes and mandate methods or objectives with their specific artifacts to pre-

vent design and implementation errors in the software layer. As a result of being unable to quantify software quality and reliability properly, both standards prescribe a development work flow consisting of several processes which are necessary to assure the development of safety-critical software components.

In this section, a comparison of the standards is carried out along the following dimensions:

- confirmation measures,
- life cycle processes, and
- artifacts.

ISO 26262 is an automotive safety standard for mass production of passenger cars. DO-178B targets software systems that are part of an "airborne system", which are certified on a per aircraft type basis. Resulting differences concerning the confirmation measures are discussed in Section 2.1.

Section 2.2 compares the life cycle assumed in ISO 26262 to that in DO-178B with particular focus on software development.

As a main distinction between the standards, ISO 26262 recommends concrete measures to ensure safe software in Part 6. DO-178B specifies objectives to be fulfilled. ISO 26262 also mentions objectives, but rather as a justification for the recommended methods. Meeting the objectives is documented as part of the artifacts produced for the Safety Case; Section 2.3 provides a detailed comparison of the artifacts required in DO-178B and ISO 26262.

2.1 Confirmation Measures

Confirmation measures are the measures required by the standard to obtain acceptance of the product as per the requirements of the standard. Where relevant regulation is in place, confirmation measures lead to certification where the issued certificate states compliance of the product with the regulations.

While DO-178B explicitly states its relevance for certification in the avionics domain, ISO 26262 does not foresee official certification measures. Taking into account the life cycles of the two standards, the finally certified airplane is approximately equivalent to the item when "released for production" [4, Part 4, Clause 11]. In both cases, the final safety assessment has been carried out based on the artifacts created during the development process; the product is signed off for use (avionics) or mass production (automotive) by an authorized person. Both standards require interaction with the assessment body as part of the confirmation measures. For lack of a real assessment body, ISO 26262 establishes a dedicated role, the *functional safety manager*. The person with this role is responsible for the correct implementation of the safety processes and measures. DO-178B requires a dedicated process for interaction with the certification authority.

Both standards state that certification or release for production can only be done if top-level safety requirements are met for the specific fully integrated product, i.e., the airplane or the vehicle. Consequently, software development level artifacts and verification records are part of the overall assessment of the product's safety; their provision can make the software "certifiable" but not "certified".

Both standards define a software level that is a function of the seriousness of the consequences of a failure of the item. ISO 26262 also includes aspects like controllability and exposure in the level. Both standards define five levels; the level with no impact of the software on safety is called QM in ISO 26262 and Level E in DO-178B. Development of this software only requires basic quality management. ISO 26262 defines the *Automotive Safety Integrity Level* (ASIL) A to ASIL D for with increasing effort for safety measures. Similarly, DO-178B defines levels D to A, with increasing effort for safety measures.

ISO 26262 defines three types of confirmation measures. The *safety audit* is used to confirm that processes are carried out as defined and required. *Confirmation reviews* verify the contents of artifacts created during development. Finally, the *functional safety assessment* verifies the correctness of the safety case. For higher ASILs, the level of independence increases but confirmation measures can for all ASILs be done by persons within the organization.

Audits and reviews are planned as part of the safety plan and recurring part of the development process. DO-178B ensures permanent reviews by means of the certification liaison process, which involves the certification authority throughout the development process. DO-178B does not define levels of independence but – for higher software lev-

els – specifies more objectives to be verified with independence. ISO 26262 takes into account organizational roles and structures to define independence. For DO-178B, independence must ensure objective evaluation (for software) and include authority to take corrective action for software quality assurance.

2.2 Life Cycle Processes

Comparing the number of process definitions in the two standards, ISO 26262 defines 44 processes, while DO-178B defines only 7. The main reasons for this significant difference are:

- DO-178B is only focused on software development whereas ISO 26262 targets item development, including system and hardware level development,
- ISO 26262 includes processes for production and operation; these are out of scope in DO-178B, and
- the method-driven approach in ISO 26262 requires a finer granularity of process definitions.

ISO 26262 groups the processes in the different parts of the standard. For the purpose of a direct comparison not all parts and processes are relevant.

Parts 3 and 4 in ISO 26262 specifically deal with system level developments and processes. These are the concept phase processes and the processes for system level development of the item. The processes defined in these two parts are mentioned in DO-178B as part of the system aspects relating to software development. However, they are not within the scope of the standard, and only provide the input to software development. Part 7 of ISO 26262 has no corresponding part in DO-178B, as the avionics standard does not assume series production of software (see also Section 2.1). Even though there is no mapping possible, the processes are relevant for an ISO 26262 development. Section 2.2.2 proposes a mapping of integral processes from DO-178B to the supporting processes of Parts 2 and 8 in ISO 26262. The integral processes in DO-178B include the certification liaison; this is discussed in Section 2.1.The software development related processes of the two standards, defined in Part 6 of ISO 26262 are compared and mapped in 2.2.3.

2.2.1 Planning Processes

Planning the software development is part of both standards. Both standards require the definition of the software life cycle. ISO 26262 provides more guidance by proposing a V-model based approach. The planning processes for both standards include the definition of development activities, safety requirements, coding standards, methods and tools for software development, and appropriate verification plans.

ISO 26262 starts the software development process by "initiation of the software development process" where the functional safety activities as part of software development are planned. Part of the planning is a tailoring of the appropriate methods recommended in the standard to match requirements from the specific ASILs. DO-178B defines the software planning process which states similar objectives. Both standards acknowledge the relevance of the hardware definition for the software development. ISO 26262 is more explicit in requiring coordination with the hardware development processes.

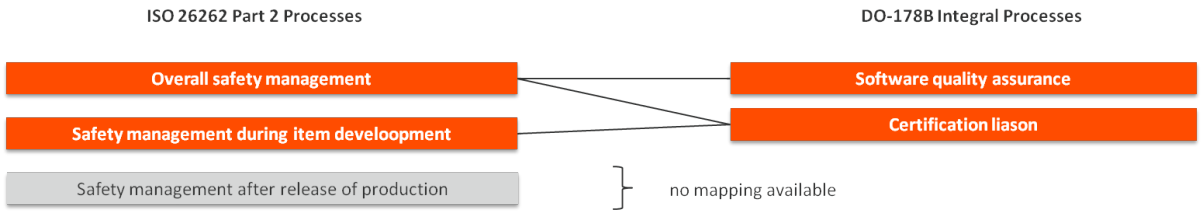
2.2.2 Integral and Supporting Processes

Integral and supporting processes are those horizontal processes that support the development of software and provide the interface to review and certification activities. While the relevant support processes for ISO 26262 are distributed over several parts of the standard, DO-178B defines four *integral processes*. Two integral processes target safety management of the software and two can be identified to be *supporting processes* in the sense of ISO 26262.

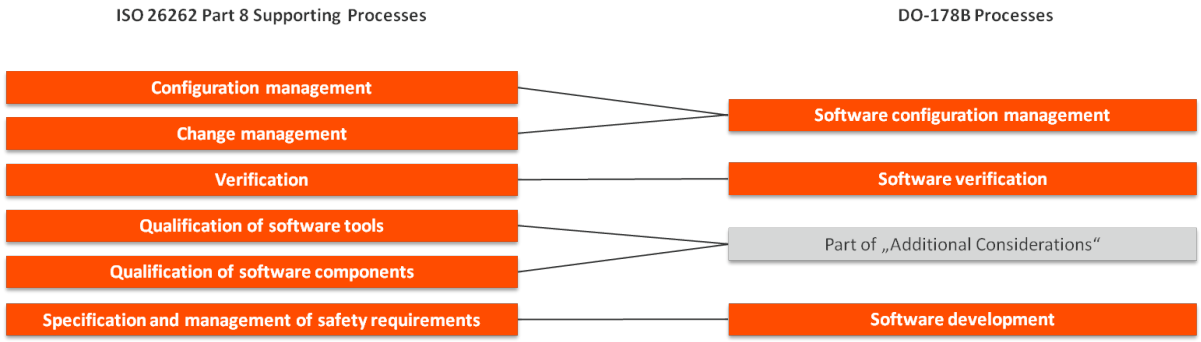
Figure 1a provides a mapping of safety management processes defined in ISO 26262 to matching processes in DO-178B. As the scope of DO-178B does not include mass production, there is no mapping to the safety management after release of production.

A mapping of supporting processes from ISO 26262 to DO-178B integral processes is provided in Figure 1b. ISO 26262 provides the processes in a finer granularity, but the objectives are similar. The figure does not contain those processes that could not be mapped; these are discussed below.

The following processes of ISO 26262 have no



(a) Safety management related processes



(b) Supporting processes

Figure 1: Mapping of supporting and integral processes from ISO 26262 to DO-178B

direct counterparts within DO-178B:

- Interfaces within distributed developments [4, Part 8, Clause 5]
- Documentation [4, Part 8, Clause 10]
- Proven in use arguments [4, Part 8, Clause 14]
- Qualification of hardware components [4, Part 8, Clause 13]

Interfaces within distributed environments in ISO 26262 have been introduced to account for the joint and distributed development of an item. By the specification of this process, which is an explicit addendum to IEC 61508, ISO 26262 accounts for the distributed development of automotive systems. The resulting Development Interface Agreement is an artifact that must be provided as basis for a safety case in a distributed development (see Section 2.3).

DO-178B requires that the outputs of the life cycle processes for previously used software should conform to the standard. Different to the approach of mapping life cycle processes, with the proven in use argument, ISO 26262 allows the re-use of components based on extensive field experience in a comparable use case. This helps integrating COTS components, provided the appropriate field data are available. Note that it may be difficult to argue for sufficient comparability of two use-cases.

The documentation process in ISO 26262 is present to develop a strategy to produce precise, structured, understandable, and maintainable documentation. The requirements for documentation of the life cycle processes in DO-178B are similar, but there is no directly comparable process for documentation creation in the avionics standard.

Finally, hardware component qualification is not present in the DO-178B, as the standard only deals with software development.

2.2.3 Software Development Processes

Software development in ISO 26262 starts with the specification of software safety requirements, after initiation of product development at software level, this is equivalent to the software planning process in DO-178B. ISO 26262 a reference phase model and subphases corresponding to the V-model depicted in Figure 2. The subphases define activ-

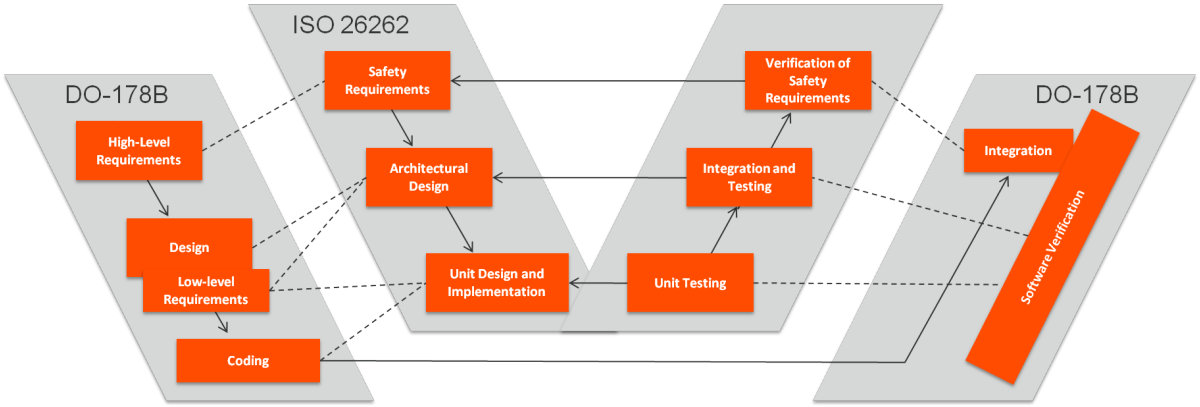


Figure 2: Software development phases in ISO 26262 and DO-178B, dashed lines map the phases

ities as part of the software development process. The figure also provides a mapping between the different subphases and activities from the two standards.

The development model, defined as part of the software development process of DO-178B describes activities suggesting a waterfall-like approach: requirements (elicitation), design, coding and integration. The activities are the basis for defining the software development cycle. The activities can be different for different software components (e.g. leaving out the implementation part for existing components). Software verification and testing is part of a separate process, which includes testing the software and verifying the implementation of the requirements on different levels. The standard does not make the explicit distinction between the software and software units; this distinction is implicit in breaking down high-level requirements to low-level requirements.

Both standards require explicit planning and tailoring of the software development process using the activities / subphases as building blocks. Therefore, a mapping of existing building blocks should be possible, despite the greater flexibility of DO-178B to define the software life cycle.

2.3 Artifacts for Software Development

Both ISO 26262 as well as DO-178B require specific artifacts to be produced during development of the product. For ISO 26262, the list of work products is a result of the tailoring process for a specific item. As software development only covers a part of item development, only those artifacts that potentially contribute to a safety case are selected for the comparison in this section. DO-178B explicitly describes the objectives of each artifact – called life cycle data in the standard – in Chapter 11, whereas the work products in ISO 26262 are defined as part of the processes generating or revising the artifacts. On item level, to issue the "release for production", the following artifacts are necessary [4, Part 4, Clause 11]:

- Functional safety assessment report
- Safety case

The functional safety assessment report contains the results of the confirmation measures for the item. All relevant work products are reviewed in the assessment. In order to clear an item for release of production, the assessment report must contain the recommendation for acceptance of the functional safety of the item. The functional safety assessment is equivalent to a certificate stating conformance to the standard. For DO-178B this artifact is left to the discretion of the certification authority.

The safety case references the relevant work products to be evaluated to prove the safety of the item. It is the basis to evaluate of safety for the item. In Part 6 of ISO 26262, the work products depicted in Figure 3 are mentioned. The figure

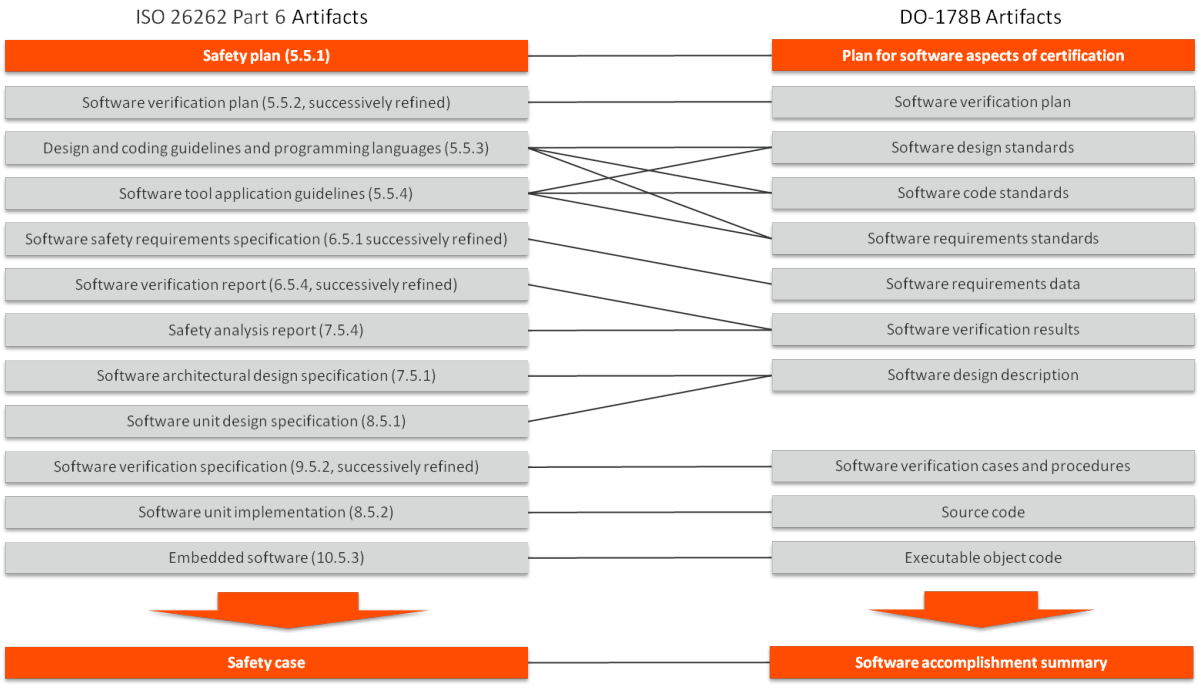


Figure 3: Artifacts from ISO 26262 for software development and a mapping to DO-178B artifacts

also provides an initial mapping of the artifacts of ISO 26262 to those of DO-178B. The work products mentioned may require input or assumptions from work products produced by other processes not included of Part 6. They may need to be part of assumptions for product development.

Figure 3 depicts the safety plan and the safety case for ISO 26262 embracing the artifacts constructed for the software. They are primary input for the confirmation measures to create the functional safety assessment report. While the plan defines interaction with the assessment body, the safety case defines the relevant input and argumentation that a specific safety level is achieved. This is similar for the respective artifacts in DO-178B.

Summing up, the artifacts between the standards map well, taking into account the objectives and activities necessary to create them. Note that this analysis does not take into account specific requirements for different software levels (ASILs), where the requirements, concerning objectives and required methods may differ for the two standards.

3 Case Study

Some current efforts in the automotive industry target integrating infotainment systems with selected safety-relevant functions on the head unit or the instrument cluster. Figure 5 depicts a typical architecture integrating Android^{TM}, and a safety-relevant function on one platform.

3.1 Approach

The high-level comparison of DO-178B and ISO 26262 in Section 2 indicates a comparability of the standards in terms of processes and artifacts. Concrete objectives and methods as part of the processes can only be provided based on a concrete use case, with concrete assumptions on a required safety level. This approach is similar to the *safety element out of context (SEooC)* approach as described in [4, Part 10]. The approach to carry out a case study is as follows:

1. Define a representative item, and document the relevant assumptions, in particular about

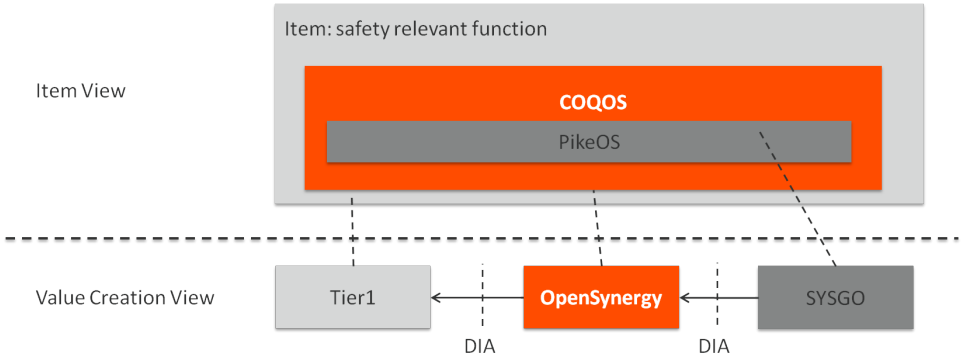


Figure 4: High level overview of a representative automotive item for the use in a case study. *DIA* is the Development Interface Agreement between the different parties in the distributed development.

functional safety requirements and required ASIL.

2. Focus on software level, tailor processes and work products to match ISO 26262, identify appropriate ways to integrate existing, certified software.

In this work a representative item, as well as corresponding safety goals are proposed as a starting point for the case study. The actual case study, based on the findings of the overall comparison of the standards is left for future work.

3.2 Item Definition and Safety Goals

The development of a safety-relevant function as depicted in Figure 4 is assumed. The different parts of the system may be developed and integrated by different suppliers. This is depicted in the value creation view in the bottom half of Figure 4.

For the concrete example a commercially available product COQOS on PikeOS, provided by OpenSynergy and SYSGO, respectively, shall be used. Interfaces and agreements between the players in the "value creation network" [2] become relevant, once the co-operation between the players is set up to carry out the development of a safety-relevant function. The main aspects of the item from the perspective of this work are:

- PikeOS, a microkernel certifiable to DO-178B by Sysgo
- COQOS, an operating system framework for automotive by OpenSynergy

PikeOS, the microkernel, is a commercial-off-the-shelf (COTS) component certifiable to DO-178B. The kernel has already successfully been certified to level B for the use on the PowerPC platform. For the certification, the PikeOS kernel provides at least separation of platform resources, health monitoring, as well as assurance of correct functioning of system services.

The case study will provide details of how this system can be integrated in a ISO 26262 setting for the development of certifiable automotive systems. COQOS, the operating system framework for automotive provided by OpenSynergy, integrates the PikeOS microkernel and provides automotive-specific features. The main features are fast boot, firewall between partitions, real-time support, inter-partition communication.

Figure 5 depicts the different types of partitions that can be present in COQOS. These are the infotainment (HMI, human machine interface) partitions, early applications, as well as safety-relevant applications, such as those developed as AUTOSAR components. For the purpose of the item, it is important that those partitions are safely separated and can only communicate according to pre-defined rules.

The use of a microkernel-based operating system can provide higher safety and security guarantees given the microkernel has been developed accordingly. The microkernel provides the pos-

Figure 5: Architecture of a representative automotive item for the use in a case study

sibility to clearly separate partitions in terms of processing time, and access to resources (e.g., I/O channels, memory) – a property called *separation*. Separation, together with other technologies such as virtualization, support for different personalities and inter-process communications, is the basis for providing the main features of COQOS mentioned above.

In a nutshell, separation is the main functional safety requirement on the kernel, for item development. Note that additional safety requirements will probably be created upon analysis of the functionality necessary for the whole item.

3.3 Assumed ASIL

Assuming the example use case being part of the instrument cluster, the required ASIL can be assumed for the purpose of the case study. ASIL estimation is based on estimating severity, exposure and controllability for the specific function. The example function on the instrument cluster can be described by the triplet (*S2*,*E4*,*C2*), which leads to ASIL B as per Table 4 in [4, Part 3].

Severity *S2* describes "severe and life threatening injuries" caused by unintended behavior of the item. Assuming that the instrument cluster only displays messages and warnings, this may already be overestimated. The class is nevertheless used here, in order to be on the safe side for later item integration. Exposure *E4* is with "high probability" for the fictional item. This is the highest value for this aspect. The value is appropriate, given that the functions of the instrument cluster are visible whenever the vehicle is started and operated.

The controllability in case of the item's failure is *C2* meaning "normally controllable". According to the definition, 90% of the drivers would be capable of avoiding specific harm in case of an item's failure. Given that the functions on the instrument cluster only provide warnings and indications, a driver should be able to neglect implausible values or displays of the function.

3.4 Discussion

From the above assumptions and the review of the two standards a complete safety case can be constructed for the case study. The functional safety requirements for COQOS and PikeOS, as well as the software level (ASIL B) in mind, the process for software development can be tailored and the artifacts in Section 2.3 created.

There are different possibilities to integrate the certified microkernel into the system. One possibility would be to qualify the software component as per ISO 26262 Part 8 Clause 12. The artifacts created for DO-178B certification will almost certainly be sufficient to support qualification of the kernel.

Another possibility could be to create the appropriate work products and processes to construct a safety case with the microkernel as another software component. The discussion in the previous sections indicate that this approach could also be possible. This approach may require the creation of more work products, such as the Development Interface Agreement. The advantage could be that the microkernel would better suit a distributed environment as foreseen for future automotive developments.

4 Conclusion

This work argues that a mapping between the automotive safety standard ISO 26262 and its avionics counterpart for software DO-178B exists. The work presented here is work in progress.

ISO 26262 development can make use of the artifacts and processes defined in DO-178B. At least the distributed development process, as well as the processes for the production and operation must newly be accounted for. As another way to integrate previously developed software, ISO 26262 provides the software qualification process or proven in use arguments.

To obtain a more definitive statement, a case study must be carried out for a concrete enough representative item; the approach for this has been outlined in this work. Besides the case study, consultancy with more experts for the different standards will be beneficial to clarify the mapping and appropriate integration of avionics software.

Acknowledgements

This work is carried out as part of the VirtuOS project. The VirtuOS project is financed by TSB Technologiestiftung Berlin – Zukunftsfonds Berlin Co-financed by the European Union – European fund for regional development.

References

[1] AUTOSAR website. www.autosar.org, 2010.

[2] R. Hamann, J. Sauler, S. Kriso, W. Grote, and J. Mössinger. Application of ISO 26262 in distributed development in reality. Technical report, SAE International, 2009.

[3] IEC. IEC 61508: Functional safety of electrical/electronic/programmable electronic safety related systems. Technical report, IEC, 1998.

[4] International Standards Organization. ISO DIS 26262: Functional safety for road vehicles. Draft International Standard ISO 26262, International Standards Organization, 2009.

[5] OpenSynergy. COQOS. www.opensynergy.com, 2010.

[6] RTCA Inc. DO-178B: Software considerations in airborne systems and equipment certification. Technical Report DO-178B, RTCA Inc., Washington, USA, 1994.

[7] SYSGO. PikeOS. www.sysgo.com, 2010.