2014 NDIA GROUND VEHICLE SYSTEMS ENGINEERING AND TECHNOLOGY SYMPOSIUM
SYSTEMS ENGINEERING (SE) TECHNICAL SESSION
AUGUST 12-14, 2014 - NOVI, MICHIGAN

A Comparative Analysis of Aviation and Ground Vehicle Software Development Standards

Kevin Crots
DornerWorks, Ltd.
Grand Rapids, MI

Paul Skentzos
DornerWorks, Ltd.
Grand Rapids, MI

Dan Bartz
Booz Allen Hamilton
Troy, MI

ABSTRACT

The integration of software into transportation systems is growing and requires the adoption of safety standards and software development systems. There are several different safety standards that could be applied based on the specific category of use. The basic methodologies used in these standards can be applied to any transportation system including Ground Based systems. This paper evaluates two different safety development standards and provides a high level comparison between a well-used standard for aviation and a more recent standard for automotive that can be applied to other transportations systems with no available standards.

INTRODUCTION

The aviation industry has long adopted standards that incorporate safety planning and safety specific process development as a larger part of the software development cycle with the adoption of DO-178 in the 1980's. The automotive industry has recently started adopting ISO 26262 as a safety driven development system that incorporate safety design into the software development process for high volume production vehicle. The use of software in vehicular transportation over the last thirty years has shown the necessity of safety oriented development standards, especially as systems become more complex with millions of lines of code and autonomous systems are being deployed in transportation systems. Highly complex software and electronic systems used in transportation whether aviation, civilian, or military all have direct impact on the safety of society every day.

There has been considerable interest in the potential benefits automated vehicle technology promises for commercial and military applications. As automated vehicle systems transfer the responsibility for the operation of multi-ton vehicles from the driver to software, these systems must be designed from the ground up as safety critical systems with sufficient levels of redundancy and fault tolerance. The adoption of these standards to both optionally manned and unmanned ground vehicle systems are applicable for ensuring that good development practices are covered to help mitigate safety and risk. This paper provides a comparison of the development standards that have been used in the aviation sector and that have transitioned to automotive vehicle development that could be applied to Ground Vehicle development.

Overview of Standards

The aviation industry has been using DO-178 for software development processes for the last 30 years. There have been several updates to this standards with the latest being DO-178C released

in 2011 by Radio Technical Commission for Aeronautics (RTCA). This standard provides a framework and guidance for both developing a software development process as well as integrating requirements based traceability and management through the process.

The DO-178 development process guidelines consist of the following steps that are used as part of the normal development cycle for software:

- Evaluation of System Impacts to Software
- Software (SW) Planning Process
- SW Design and Coding Guidelines
- SW Requirements Process
- SW Architecture and Design
- SW Integration and Development
- SW Unit Verification
- SW Integration Verification
- SW Artifacts and Review

These steps produce artifact outputs that can then be used as part of the certification process of the software as a part of a larger system.

The standard ISO26262 is a standard that was released in 2012 that is targeted for high volume production of automobiles with a maximum gross weight of 3500kg. It is derived from the IEC61508 standard which is a broader industrial safety standard. The ISO26262 standard is intended to focus on vehicles that have electronic systems that could impact the safety of a vehicle. The standard has several parts that include safety concept, system safety, hardware and software development for safety, and then verification the requirements and safety at all of these levels. The software standards specifically, and the focus here, is found in part ISO26262-6.

The ISO26262 process development guidelines are a more specific set of development procedures than DO-178. The procedures follow the development cycle below and are intended to flow in parallel but be integrated with a standard software development cycle. The first element listed is from system components defined in section 4 of ISO26262.

- System Impacts to Software (Section 4)
- Software Requirements and Safety Goals
- Software Unit Architecture
- Unit Detailed Design and Analysis
- Unit Software Coding
- Unit Testing and Verification
- Integration and Testing
- Verification of Safety Targets and Objectives

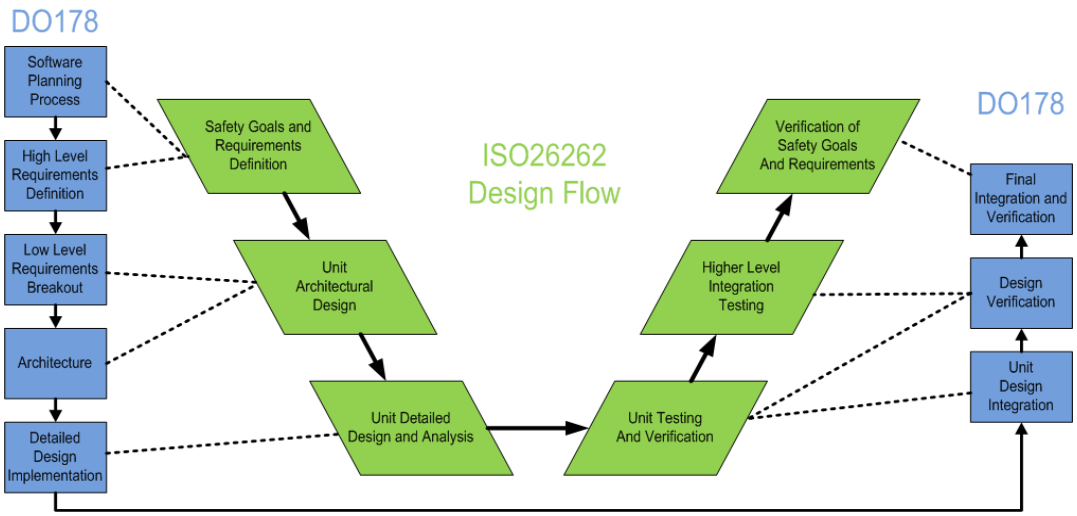

Figure 1

As shown in the comparison so far and in Figure 1, the two standards have similar process flow that can be aligned. Each of the standards have their distinctions in terminology and outputs while still following the same methodology and processes that can be used with vehicle software development from a safety perspective for both operator controlled and autonomous vehicle systems. However, application of the DO-178 and ISO26262 standards to larger and autonomous ground based vehicles must bear in mind the additional risk and impacts of non-operator controlled systems as well as larger and heavier systems which present additional safety risk. The safety oriented methodology of the standards is applicable for all systems however as they are similar however the details for each specific safety case must be evaluated for the each specific transportation system and environment.

Adoption of Aviation Function Safety SW to Automotive Processes

The developed culture of using software safety development processes for decades in aviation presents a strong background of learning and application for vehicle software development. Lessons learned, software safety life cycle development, and verification methods have shown to be effective at reducing risk in the aviation industry. These lessons can easily be transferred to automotive and autonomous vehicle systems that are shifting to primarily electronic and software driven control platforms.

Software Planning

The initial part of the DO-178 software development process is the actual planning of the process itself. The detailing of the actual software development cycle needs to incorporate a complete software life cycle, definition of relationships between the design processes and feedback mechanisms, the software development environment, and development standards. The design of these development plans for the software development lead to the beginning of artifacts that are used throughout the full development cycle and updated through each step. Definition of these standards and processes put in place the infrastructure for a safety system to be managed and followed. Preliminary artifacts can include:

- Plan for Software Aspects of Certification
- Software Development Plan
- Software Verification Plan
- Software Configuration Management Plan
- Software QA Plan
- Software Requirements Standards
- Software Design Standards
- Software Code Standards

The ISO26262-6 standard follows aviation methodology but has a more defined software development cycle. Many items including the items 3-8 in the list above are applicable to the automotive standard to meet the necessary safety schemes of the specific product being developed. ISO26262-6 has a more pre-defined plan for the actual software development itself compared to DO-178. The standard uses a V approach to the development of the planning and for definition of the process. A V development approach starts at the stop with requirements from the system level and cascades down through design while the return side is verification and integration back up to the top of the system interconnections.

Safety and Requirements Definition

Following the planning process, for DO-178 and ISO26262, already defined system requirements flow down into the requirements definition for the software level requirements. For DO-178 the system level requirements from sources outside the DO-178 development process. The actual requirements need to be evaluated for consistency,

feedback into the higher level system, traceability, and ability to verify full functionality and safety. The individual requirements also need to take into account the needed safety requirements that have been defined at a system level. The impact of each requirement on the system safety must be analyzed as well as the impact to other requirements. The DO-178 evaluation of safety or hazard classification is performed at the requirements stage. Within the ISO26262 standard, the safety requirements are cascaded from section 3 where the safety concept is defined and section 4 where the system is defined and split out into hardware and software. The outputs for both of the standards are requirements documentation as well as updates to any software plans and verification plans developed as part of the original software planning.

The cascading of system requirements and a software plan into functional and safety requirements lower level requirements is a critical piece of any design cycle for integrated software and safety. With the development of autonomous and ground vehicles accelerating, it is critical that key steps in processes such as these be included and a full vetting of new potential hazards with a complete reliance on electronics systems, vehicles that pose a larger safety hazard in certain situations, and the complexity of integration with external systems that can impact control of the vehicle.

Hazard Classification

When defining the hazard classification that is used in the requirements phase, there are different levels of safety criticality that impact how the software is designed, diagnosed, and verified. Table 1 highlights these differences as the classification levels are reversed depending on the standard. The ratings impact the requirements that must be put with each software unit in terms of design, fault detection and prevention, and verification testing.

Safety Impact	DO-178	ISO26262
Catastrophic	A	D
Hazardous	B	C
Major	C	B
Minor	D	A
No Effect	E	Quality Managed

Table 1

ISO 26262 introduces the concept of controllability to hazard classification. This incorporates the ability for the operator to mitigate an accident if the system fails. Autonomous vehicles add a new dimension of safety rankings and categorization because there is no operator to provide this control. The safety capabilities that could be utilized for operator interaction to put the vehicle in a safe state are removed. A new classification system would need to be developed to evaluate the failure rate, the impact to life and limb, and the proximity for damage to a larger eco-system around the vehicle. The base principles of evaluation through still apply from aviation and automotive for looking at the safety aspects. Furthermore, current standards are not designed to supposed multi-vehicle cooperative systems, which are a major use case for both commercial and automated vehicle systems.

Software Architecture, Design, and Implementation

After requirements the DO-178 process uses a cascading system for the actual development flow of the software. The process begins with the high level architecture based on the requirements that were defined at an earlier stage. After the architecture is developed, the process transitions into developing low level requirements for each unit of the architecture that needs to be broken down into smaller units. These smaller units are then evaluated for a hazard classification as a smaller unit based upon the impact to the larger architecture. After the lower level requirements are developed, source code can be generated that is traced to that specific requirement.

ISO26262-6 follows a similar methodology of cascading the requirements into an architecture and design. The high level safety requirements are moved into an architectural design and then into a software unit for actual design, implementation, analysis, and simulation. Each of the software units will have its own safety rating that must be taken into account during the design, documentation and verification of the unit.

The outputs of these two standards are source code, design description details, application issues related to safety that need to be addressed, and any updates to the verification plans that have been added during the design. The source code needs to be reviewed to make sure that the code is performing its intended functionality within the scope of the requirements. Also items such as diagnostic coverage, fault mitigation, external fault impact, and outside intervention strategies must be evaluated during the architecture and design phase to ensure that the software meets the safety goals and requirements.

Verification-Unit and Integration Level

Development Verification is a necessary part of the process to verify the software does what it is intended to do. The evaluation of the outputs in DO-178 is done through verification at multiple levels including

- Verification that the software functionality meet the system requirements
- Verification that the software design and architecture meet the software requirements and standards
- Verify that the source code outputs meet the standards and architecture
- Testing of the software with outputs that prove out the functionality and safety
- Verification of the testing outputs with tracing.

Several of these items require personnel independence from the original developer for review and verification. This independence allows for a thorough review and better coverage of potential issues that might be seen in the results.

The actual verification and analysis plan is developed from the planning phase and as the design becomes more detailed, additional details are added to the plan. The plan must incorporate traceability to show that all of the requirements and units are fully tested. The amount of testing is determined by safety requirements as well as the requirements determined to satisfy confidence in the software. Testing is to be done starting at the unit level and then proceeding up to a larger level of integration until the full system is tested and verified.

For the ISO26262 standard, the verification procedures are similar to aviation. The verification plan is written early and then details are added as the design proceeds. The testing begins at the unit level. Certain levels of testing are required based on the safety level and the probability of potential failure models. This standard also requires independence in certain areas for review. The output of both of these standards verification testing reports documenting the test states and results as well as verification reports.

The verification of the architecture, unit level design and implementation, and upper level integration in meeting the safety and functional requirements has been critical for aviation and automotive but takes on a new dimension with autonomous and larger vehicles. When a system is autonomous, the potential external interactions and internal unknowns must be thoroughly reviewed and vetted to verify that proper verification coverage has been obtained.

Artifacts and Governing Bodies

If followed properly, both DO-178 and ISO-2626 will produce artifacts that can be reviewed, by the governing body as a larger certification review. The FAA requires the use of DO-178 for developing software but following the process does not guarantee certification. The documents

that are used are evaluated as part of a larger certification process for the whole aircraft. For automotive, there are no external governing bodies. The OEM manufactures are responsible for the safety of the vehicle system they produce. For other transportation systems, it is necessary to identify the authority that has the responsibility for reviewing and approving the software design and test.

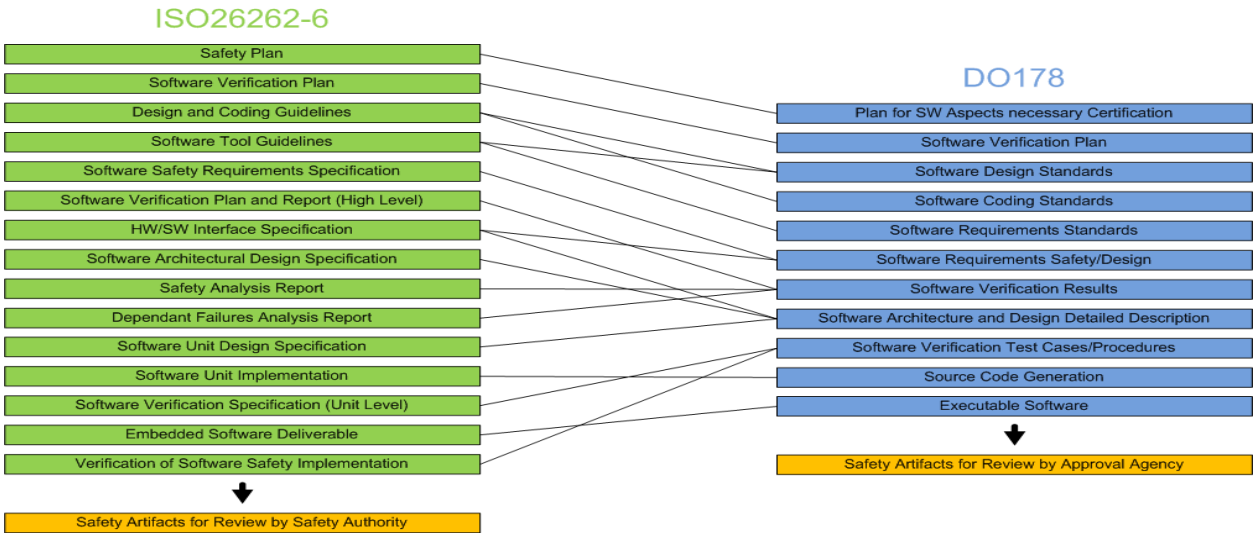

Figure 2

Figure 2 identifies a comparison of the artifacts that are developed from the two standards.

Support Tools

For each of the development standards, any supporting tools and processes that are used must be evaluated and certified for repeatability and known outputs of the tools. Quality Assurance systems, Tool Qualification Plans, Change Management, Standards, and Certification Liaison processes are all part of this supporting tool set. Figure 3 highlights the comparisons between the two standards.

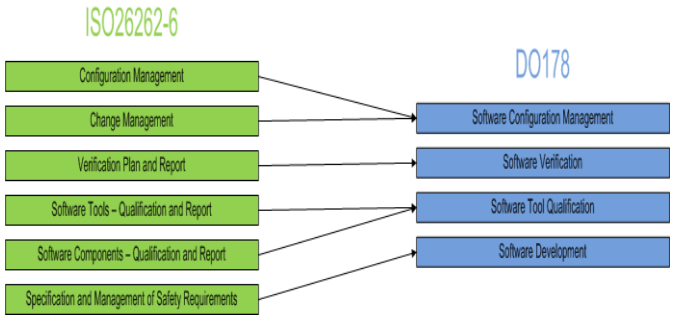

Figure 3

Real Life Software Process Integration

A demonstration of a current software development process following the DO-178 will be presented. The example will be the development of an ARLX Hypervisor built on the Zen platform which allows isolation from a safety perspective for multiple OS domains on a single hardware platform. This Hypervisor must follow the DO-178 standard because it is being used as a safety isolation device for critical pieces of software running a single hardware platform where other non-safety critical items might be running also.

The first phase was planning. The planning addresses items including software development platforms, software processes, quality assurance programs, change control process, and other coding and software standards. Outputs such as the following were produced such as Figure 4 which shows development sequences and Figure 5 which shows proposed architectures.

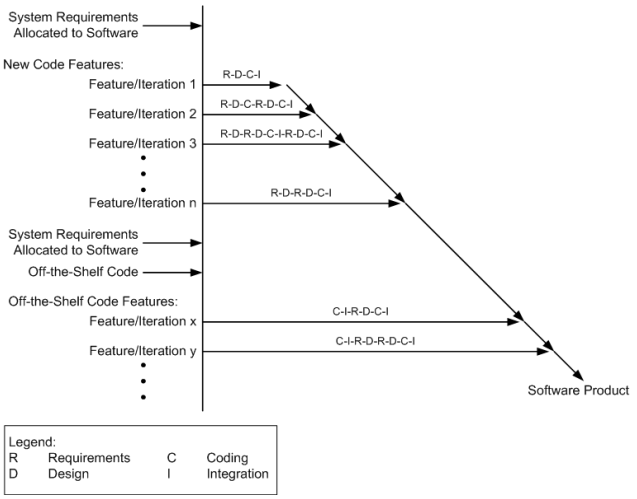

Figure 4

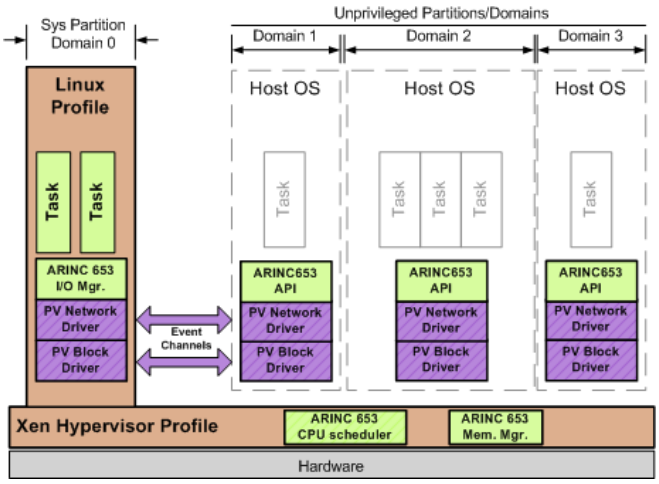

Figure 5

The next phase was the requirements gather and definition. The system architecture was designed in the previous phase but the detailed components were broken down into requirements documents for items such as configuration requirements, data integrity requirements, inter-partition communications, memory partitioning, and I/O requirements. A requirements document and a verification document were created for each of these sections within the design. Each of these higher level requirements documents were broken down further into lower level requirements and a safety analysis was performed for each requirement, the impact, and necessary steps to address design concerns related to the safety rating.

The next step was the coding which commenced for the development of the actual software. The safe and secure hypervisor platform that is being developed is based on an open-source hypervisor already with additional infrastructure to support safety critical systems and secure systems running on the hardware. The additional software components that are required for the safety and security were developed and coded as part of this development platform.

The last step was the verification step and artifact generation. The components that were developed were tested against the original test plan and the results generated. These results were reviewed by the team both internal and for the external customer as part of the review process. The artifacts from each of these phases were put together in a set of documents which thoroughly detailed out the design and all of the evidence to show that the design meets the functional and safety requirements.

Summary

Automated vehicles promise significant benefits due to the removal of the operator. The removal of the operator, however, means that safety of these systems will depend on software to provide this safety functionality. Safety relevant software for future ground vehicle systems, both civilian and military, could use the standards highlighted as a software development guidelines. The applications of the development of detailed initial plans, safety driven requirements, design around those requirements, and the verification process are applicable regardless as to the specific standards. Good methodology and planned process development allow for the development of robust systems and software that minimize risk to people in the real world.

REFERENCES

[1] International Standards Organization, "ISO26262 – Road Vehicles Functional Safety" Sections 1,3,4,6 , 2011 (ISO 26262:2011(E)

[2] RTCA, Inc, "Software Considerations in Airborne Systems and Equipment Certification", December 13, 2011 (RTCA DO-178C)

[3] Matthias Gerlach, "Can Cars Fly? From Avionics to Automotive – Comparability of Domain Specific Safety Standards", VirtualOS Embedded World, 2011.