Proceedings of the International Multiconference on
Computer Science and Information Technology, pp. 665 – 672

ISBN 978-83-60810-14-9
ISSN 1896-7094

Software Certification for Safety-Critical Systems: A Status Report

Andrew Kornecki
Dept. of Computer and Software Engineering
Embry-Riddle Aeronautical University
Daytona Beach, FL 32614, USA
kornecka@erau.edu

Janusz Zalewski
Department of Computer Science
Florida Gulf Coast University
Fort Myers, FL 33965-6565, USA
zalewski@fgcu.edu

***Abstract*—This paper presents an overview and the role of certification in safety-critical computer systems focusing on software and hardware use in the domain of civil aviation. It discusses certification activities according to RTCA DO-178B "Software Considerations in Airborne Systems and Equipment Certification" and RTCA DO-254 "Design Assurance Guidance for Airborne Electronic Hardware." Specifically, certification issues in real-time operating systems, programming languages, software development tools, complex electronic hardware and tool qualification are discussed. Results of an independent industry survey done by the authors are also presented.**

I. Introduction

CERTIFICATION is the hot issue in many industries that rely on the use of computers and software in embedded systems that control safety-critical equipment. The term "certification" in software engineering is typically associated with three meanings: certifying product, process, or personnel. Product and process certification are the most challenging in developing software for real-time safety critical systems, such as flight control and traffic control, road vehicles, railway interchanges, nuclear facilities, medical equipment and implanted devices, etc. These are systems that operate under strict timing requirements and may cause significant damage or loss of life, if not operating properly. Therefore, the society has to protect itself, and governments and engineering societies initiated establishing standards and guidelines for computer system developers to follow them in designing such systems in several regulated industries, including aerospace, avionics, automotive, medical, nuclear, railways, and others.

Consequently, the U.S. government and international agencies that regulate respective industries have issued a number of standards, guidelines, and reports related to certification and/or other aspects of software assurance, such as licensing, qualification, or validation, in their specific areas of interest. Two such guidance documents for civil aviation, DO-178B [1] and DO-254 [2], developed by RTCA, Inc., describe the conditions for assurance in designing software and electronic hardware in airborne systems. The guidelines are adopted by the U.S. Federal Aviation Administration (FAA) and the European EUROCAE, as mandatory for design and implementation of airborne systems.

In this paper we present an overview of current practices in civil aviation industry and discuss issues related to certification of software and hardware to meet the guidance requirements. Section 2 discusses the role of guidance in certification, and sections 3 and 4 review the certification issues according to DO-178B and DO-254, respectively. Section 5 provides some conclusions.

II. The Role of Standards in Certification

The RTCA, Inc., previously known as the Radio-Telecommunication Committee for Aviation, is a non-profit corporation formed to advance the art and science of aviation and aviation electronic systems for the benefit of the public. The main RTCA function is to act as a Federal Advisory Committee to develop consensus-based recommendations on aviation issues, which are used as the foundation for Federal Aviation Administration Technical Standard Orders controlling the certification of aviation systems.

In 1980, the RTCA, convened a special committee (SC-145) to establish guidelines for developing airborne systems and equipment. They produced a report, "Software Considerations in Airborne Systems and Equipment Certification," which was subsequently approved by the RTCA Executive Committee and published in January 1982 as the RTCA document DO-178. After gaining further experience in airborne system certification, the RTCA decided to revise the previous document. Another committee (SC-152) drafted DO-178A, which was published in 1985. Due to rapid advances in technology, the RTCA established a new committee (SC-167) in 1989. Its goal was to update, as needed, DO-178A. SC-167 focused on five major areas: (1) Documentation Integration and Production, (2) System Issues, (3) Software Development, (4) Software Verification, and (5) Software Configuration Management and Software Quality Assurance. The resulting document, DO-178B, provides guidelines for these areas [1].

RTCA/EUROCAE DO-254/ED-80 [2] was released in 2000, addressing design assurance for complex electronic hardware. The guidance is applicable to a wide range of hardware devices, ranging from integrated technology hybrid and multi-chip components, to custom programmable microcoded components, to circuit board assemblies (CBA), to en-

The presented work was supported in part by the Aviation Airworthiness Center of Excellence under contract DTFACT-07-C-00010 sponsored by the FAA. Findings contained herein are not necessarily those of the FAA.

tire line replaceable unit (LRU). This guidance also addresses the issue of COTS components. The document's appendices provide guidance for data to be submitted, including: independence and control data category based on the assigned assurance level, description of the functional failure path analysis (FFPA) method applicable to hardware with Design Assurance Levels (DAL) A and B, and discussion of additional assurance techniques, such as formal methods to support and verify analysis results.

III. Software Certification according to DO-178B

There are three essential categories of software that impact the certification process, due to their different functionality: real-time operating systems, programming languages (with their compilers), and development tools.

A. Real-Time Operating Systems

There is an evident trend to adopt the Real-Time Operating System (RTOS) kernels to increasing scrutiny of regulatory demands. The vendors have quickly "jumped on the bandwagon" and attempted to comply with requirements of DO-178B, claiming certifiability. This includes VxWorks from Wind River Systems [3-5], as well as LynxOS from LynuxWorks, Integrity from Green Hills Software, Linux and RTLinux, RTEMS and microC.

Romanski reports [3] on certification attempts of VxWorks that started in 1999. At the start of the project, the specifications, documentation, and source code were all analyzed to determine which features need to be removed or changed to support the certification. The analysis showed that the core OS could be certified and many of the support libraries could be included as well, with some restrictions, for example on memory allocation/deallocation functions. The process was largely automated, with a database and CD-ROM materials deliverable to the auditors. Further, Fachet [4] reports on the VxWorks certification process to meet the criteria of IEC 61508, and Parkinson and Kinnan [5] describe the entire development platform for a specific version of the kernel VxWorks 653 to be used in the integrated modular avionics.

Not much information, except articles in trade magazines, is available on other real-time kernels. Applying the definition of certification as "procedure by which a third-party gives written assurance that a product, process or service conforms to specified requirements", Moraes et al. [6] use the risk assessment technique FMEA (Failure Mode and Effect Analysis) to create a metric and analyze data for two kernels RTLinux and RTEMS. The analysis shows that if the threshold to certify the software is set to an estimated risk lower than 2.5%, only RTEMS would be certified.

Interestingly, a well described process of selecting an RTOS according to DO-178B guidelines [7] led to a choice of microC/OS kernel, a relatively unknown although well documented RTOS, available for many years but not much advertised. Verification of this RTOS has been contracted to an independent organization and all requirements-based tests have been completed in 2003.

B. Programming Languages

A similar trend among vendors is visible in the area of programming languages and compilers. In an earlier article, Halang and Zalewski [8] present an overview of programming languages for use in safety-related applications up to 2002, focusing on PEARL, originated and predominantly used in Germany. Their observation with respect to DO-178B and other standards is that " because verification is the main prerequisite to enable the certification of larger software-based solutions, only serious improvements aiming to support the process of program verification will be a step in the right direction."

There are essentially three contenders among languages used in safety-critical systems: Ada, C/C++ and Java, for which DO-178B certifiability is claimed. Due to limited space, we only address Ada and C/C++. The most advanced in this respect seems to be Ada, whose certification attempts go back to the eighties, with roots in compiler validation [9].

Ada and Compiler Certification. Santhanam [10] answers the question, what does it mean to qualify a compiler tool suite per DO-178B requirements, and lists the requirements on the object code and the development process, estimating the overwhelming cost of providing evidence. Therefore, defensive techniques are advocated, to assure confidence in the compiler correctness with the use of assertions, optimizations turned off, no suppression of run-time checks, avoidance of nested subprograms, etc.

Features of the object model of Ada 2005 are claimed to be "well suited for applications that have to meet certification at various levels" [11]. It meets the safety requirement, which means that programmers are able "to write programs with high assurance that their execution does not introduce hazards" [12], in order "to allow the system to be certified against safety standards", such as DO-178B. However, the common opinion, expressed by the same authors, who actually developed compilers, is that compilers "are far too complex to be themselves certified" [11]-[12].

One version of Ada, which makes use of its severely limited subset, named SPARK, seems to have gained some popularity in safety-critical applications, because of the existence of its formal definition. Amey et al. [13] report on multiple applications of SPARK in industry, including one to the DO-178B Level A.

The C/C++ Certification Issues. In the C/C++ world, there have not been many reports on the successful uses of these languages in safety-critical applications that would pass or be aimed at any certification efforts. The languages are being widely criticized for having features not necessarily suitable for safety-critical systems.

Hatton [14] gave an overview of safer C subsets and MISRA C, in particular, following his crusade towards make C a safer language. His premise was that "C is the perfect language for non-controversial safer subsetting as it is known to suffer from a number of potential fault modes and the fault modes are very well understood in general." He analyzed the standards with respect to style related rules, divided further into rules based on "folklore" and those based on known failures. He observes "MISRA C does not address all known

fault modes, and does not incorporate the full range of analysis checks that it might."

Despite the enormous popularity of C++, the number of C++ applications in avionics is relatively low, perhaps due to the multitude of known language problems. Subbiah and Nagaraj [15] report on the issues with C++ certification for avionics systems, focusing on structural coverage, whose intent is "to ensure that all output of the compiler is tested during the execution of the requirement-based tests, so as to preclude the possibility that some instruction or data item produced by the compiler is first depended upon during operation."

C.Software Development Tools

Regarding the use of tools, the FAA recently released a comprehensive report by the current authors on " Assessment of Software Development Tools for Saf ety-Critical Real-Time Systems" [16], which has been summarized in [17] and briefed in [18] regarding tool qualification. The experimental part of this work involved collecting data from the usage of six software design tools (as opposed to verification tools [19]), in a small-scale software development project, regarding four software quality criteria. Assuming these criteria were direct metrics of quality, the following specific measures to evaluate them were defined and used in the experiments:

- usability measured as development effort (in hours)
- functionality measured via the questionnaire (on a 0–5 point scale)
- efficiency measured as code size (in Lines of Code, (LOC))
- traceability measured by manual tracking (in number of defects).
- collection and analysis of results.

Since then, a good number of articles have been written on tool verification, qualification and certification attempts. Regarding software, the tool qualification process must address the requirements of the DO-178B. In particular, the decision must be made, whether tool qualification is necessary (see Fig. 1).

A tool is categorized as the development tool, if it can insert an error in the airborne system, or as the verification tool, if it may only fail to detect an error. In the following, we try to cover issues related to software verification tools.

For verification tool qualification, several interesting papers have been published in the last few years. As Dewar and Brosgol [20] point out in their discussion of static analysis tools for safety certification, a tool as fundamental as the compiler can be certainly treated as a development tool, but also as a verification tool, since compilers "often perform much more extensive tasks of program analysis." As a perfect counterexample they refer to the Spark's Examiner, which is not a usual kind of compiler, because it does not generate code at all. It is only used for checking the program, nevertheless is a part of a software development process. Furthermore, they ask the question should the tools "be certified with the same rigorous approach that is used for safety-critical applications?" Their answer is that this is not practical, and they support this view by stating that even "the compilers themselves are out of reach for formal safety certification, because of their inherent complexity."

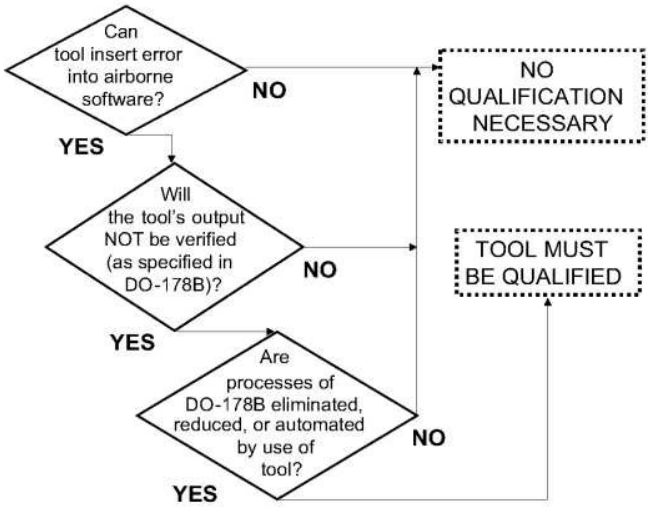

Fig. 1 Tool qualification conditions according to DO-178B [1].

Dewar supports this view in another article [21], elaborating more on the tools for static analysis of such properties as schedulability, worst-case timing, freedom from race conditions, freedom from side effects, etc. He also offers his views on the use of testing, object-oriented programming, dynamic dispatching, and other issues in developing safety-critical systems. He elaborates on the role of the Designated Engineering Representatives (DER's), whose job is to work with software development companies and the certification authorities on the qualification and certification issues, stating that DER's "are the *building inspectors* of the software engineering industry."

In another article, Santhanam [22] describes a toolset called Test Set Editor (TSE), which automates the compiler testing process and working in combination with the Excel spreadsheet and the homegrown scripts in Tcl/tk significantly contribute to cost savings in constructing structural tests to satisfy FAA certification requirements.

A recent FAA report [19] provides an overview of the verification tools available up to the time of report's publication. One tool not covered in this report, Astrée, is described in [23]. It is a parametric, abstract interpretation based, static analyzer that aims at proving the absence of run-time errors in safety-critical avionics software written in C. The authors, representing Airbus, claim that they succeeded on using the tool on a real-size program "as is", without altering or adjusting it before the analysis. Other issues addressed with this tool, although not described in the paper, include: assessment of worst-case execution time, safe memory use, and precision and stability of floating-point computations. In all that, automatically generated code should be subjected to the same verification and validation techniques as hand-written code.

It may be also worth noting that all established tool vendors have been addressing the DO-178B issues for some time now. One such interesting example is McCabe Software [24]. Their document provides a summary of McCabe IQ tool functionality and explains how the tool can be used

to support the DO-178B guidelines. Several other vendors do the same, and the current list of safety-critical software tools can be found on the web [25].

IV. Certification According to DO-254

A. Circuitry Compliance with DO-254

General Issues . With the progress of microelectronic technologies, the avionics hardware is typically custom generated using programmable logic devices. Field Programmable Logic Arrays (FPGA) and Application Specific Integrated Circuits (ASIC) are two leading implementation technologies. More often the devices include also components containing Intellectual Property (IP) chips with dedicated algorithms or custom made solutions resembling general purpose embedded microprocessor's functionality. All this caused an emergence of RTCA document DO-254 [2], which deals with safety assurance for hardware used in avionics and can be used for other safety-critical applications.

What also contributed to the origins of DO-254 is the fact that avionics companies and designers, facing the rigors of DO-178B requirements, began moving device functionality from software to hardware [26]. As reported by Cole and Beeby in 2004 [27], "There are several schemes that have been used by some to take advantage of a current loophole that allows airborne software functionality to be embedded in firmware or programmable devices. This loophole affectively sidesteps the need to adhere to DO-178B as a software standard." Thus, a new document was introduced that forms the basis for certification of complex electronic hardware, by identifying design lifecycle process, characterizing the objectives, and offering means of complying with certification requirements. The Advisory Circular published subsequently by the FAA [28] clarifies the applicability of DO-254 to custom microcoded components, such as ASIC, PLD, FPGA, and similar. In this section, we discuss recent approaches to hardware certification according to DO-254 covered in the literature.

Miner et al. [29] considered compliance with DO-254, before even the guidance was officially released. In a joint project with the FAA, NASA Langley was developing hardware to gain understanding of the document and to generate an example for training. A core subsystem of the Scalable Processor-Independent Design for Electromagnetic Resilience (SPIDER) was selected for this case study.

Hilderman and Baghai [26] offer an advice to manufacturers to map their existing development processes to those of DO-254. The strategy they recommend is "to focus on ensuring correctness at the conceptual design stage and then preserve the design integrity" as one proceeds through the development stages. Each individual vendor or designer faces multiple specific design problems that must be addressed to meet the DO-254 requirements. How they proceed depends on the vendor and the type of problem.

In the white paper of the DO-254 Users Group [30], Baghai and Burgaud offer a package including the following items designed to assist in the qualification process:

- The processes documents, that help define, benchmark and improve the industrial design, verification, validation, and quality assurance processes
- The quality assurance checklists, for reviews and audits, ensuring that each project is compliant with the defined industrial process
- The tools for requirements management and traceability, checking compliance of HDL code with coding standards, HDL code verification, and test suite optimization
- The tools integration into the industrial process, until their qualification (interfaces, report generation for a certification audit, trainings, tools assessment, etc.), and the DO-254 TRAINING by consulting partners.

Cole and Beeby [27] studied DO-254 compliance for graphic processors, considered common off-the-shelf components (COTS), and proposed a multiphase approach to meet DO-254 requirements:

- Provision of a DO-254 COTS data pack to support the use of a given electronic part.
- Provision of a DO-254 compliance statement.
- Process improvement and further analysis.
- Ongoing support for new parts and processes.

Glazebook [31] discussed certification according to DO-254 in the British context, especially the 26 data items listed in the standard as the compliance suite, of which four are required for submission: (a) Plan for Hardware Aspects of Certification; (b) Hardware Verification Plan; (c) Top Level Drawings; and (d) Hardware Accomplishment Summary. He made eight recommendations summarized in the paper.

Barco-Siles S.A. [32] report on the way the company deals with increasing demands related to implementing DO254 causing non-negligible cost, but bringing some advantages. The guidance obliges the supplier to analyze in detail processes, methodologies and tools and to apply a rigorous quality assurance. It also allows the supplier to adapt its set of internal processes to the design assurance level targeted, to optimize efforts while requiring the subcontractor to respect a structured development processes. The resulting products have improved quality and the development cycles are optimized. Verification is focused on design errors, and effort and resources are better distributed. Applying the DO254 gives the assurance that the applicant can obtain from the subcontractor a good level of quality, good documentation, and the ability to reuse the design, if necessary.

When the complexity of designs increases, it is more and more difficult to verify the correctness of circuits and thus their compliance with the specifications. As Karlsson and Forsberg point out [33], "…tests and deterministic analysis must demonstrate correct operation under all combinations and permutations of conditions down to the gate level of the device." To comply with the requirements of DO-254 they developed a design strategy that relies on a semi-formal solution, a hybrid of static and dynamic assertion based verification. They believe that by such independent assessment using their method of tool outputs, the tool qualification will become unnecessary.

EDA Industry Views . Chip and board manufacturers are eager to comply with DO-254, due to their concerns about the market share. Since compliance with the guidance is considered a technological advantage, most of the vendors began changing their development processes towards meeting the DO-254 criteria. Several companies announced their readiness to comply with certification requirements.

Mentor Graphics is particularly aggressive in providing compliance of their products with DO-254. Lange and Boer [34] give an overview of functional hardware verification methodologies, as a part of the design process. They observe that the verification techniques that served well the designs 10-15 years ago are no longer adequate due to a tremendous increase in design complexity and integration. As a consequence, design verification has become a limiting factor in safety-critical systems, with respect to such issues as: complexity, concurrency and metastability. Latest verification techniques are described that handle problems such as state explosion, design traceability and the effectiveness of coverage.

Advanced Verification Methodology (AVM), consisting of constraint random test generation, a total coverage model, design intent specification, and formal model checking, described in [35], has been used on a practical design of FPGA based DMA engine at Rockwell-Collins. The approach based on an open source Transaction Level Modeling (TLM) class library, is vendor neutral and supports SystemVerilog and SystemC standard languages. Due to the open source nature AVM allows code inspections that may be required for certification. Although the project has not been fully completed at the time of this writing, it is believed that AVM helps not only demonstrate that the DO-254 guidelines are followed, but also assists in shortening design cycles.

These verification steps/techniques must be performed in concert with the RTL design, ultimately leading to automatic circuit synthesis [36]. Since automatic synthesis and conversion to gate-level designs is often done with optimizations by the hardware design tools, it may be counterproductive in safety-critical designs, which mandate strict adherence to the guidance. DO-254 defines tool qualification, "to ensure that tools used to design and verify hardware perform to an acceptable level of confidence on the target project." The paper comments on three methods of DO-254 allowed tool assessment: relevant history, independent output assessment, and tool qualification. Since proving relevant history and qualifying the tool are both tedious and expensive processes, requiring the submittal of data, which may not be easily available, the paper suggests the product assessment route to demonstrate that "the hardware item must be thoroughly verified against the functional requirements", thus, the independent tool assessment is not necessary. In the opinion of current authors, the tool output is still an abstract entity, not the hardware item yet, and may contain errors that cannot be detected during verification.

Lee and Dewey [37] shed more light on meeting DO-254 guidance in a form acceptable to the DER, by explicitly proposing:

- requirements management and tracking, with the use of such tools as Reqtify or DOORS
- Register Transfer Level (RTL) code validation, with an automated method to measure RTL to a company standard
- verification process assurance, with the use of AVM, and
- producing design documentation, from requirements, to the RTL code, to the bit streams or Graphic Data System (GDS) II file format.

The mindset of the paper is that "DO-254 is not a burden but a set of guides that helps standardize hardware systems assurance, making flight systems safe."

Aldec and Actel, working in alliance, published some information on their efforts towards making their products DO-254 certifiable. Sysenko and Pragasam [38] outlined their process for airborne systems design assurance which relies on the verification methodology called Hardware Embedded Simulation (HES) and follows two traditional steps: RTL simulation and gate-level simulation. It is a hardware-software simulation platform driven by software that facilitates the implementation of the design in a reconfigurable hardware, such as an FPGA, and then verification of the design functions. Earlier, Land and Bryant [39] presented more details on the process, with MIL-STD-1553 bus chip design as an example to comply with DO-254.

Lundquist in his thesis [40] looked at the problems that arise when trying to DO-254 certify system-on-chip solutions. Since more than 700 Actel FPGAs are used in the Airbus A380 commercial airliner, the Actel Fusion FPGA chip with integrated analog and digital functionality was tested according to the verification guidance. The results have shown that a certification procedure for a standard non-embedded FPGA based safety critical system is possible. However, the question of how these embedded chips could pass certification to be used in safety-critical systems has not been answered.

B.Tool Certification against DO-254

Since the growing complexity of electronics hardware requires the use of automatic software tools, the DO-254 document also includes a section on tool qualification. It distinguishes between design tools, which can introduce errors into the product, and verification tools, which do not introduce errors into the product but may fail detecting errors in the product. The qualification process tool vendors have to comply with is shown in Figure 2.

Several vendors recently began dealing with tool qualification. Aldec [41] used a sample design of a system containing two connected boards: Aldec HES board (HES-3X3000EX) generating stimuli and collecting results for Design Under Test (DUT) and the second user designed board. The verification process contained three independent stages: simulation, verification, and comparison.

Lange [42] addresses circuit metastability in the context of DO-254 tool certification. Metastability describes what happens in digital circuits when the clock and data inputs of a flip-flop change values at approximately the same time. This

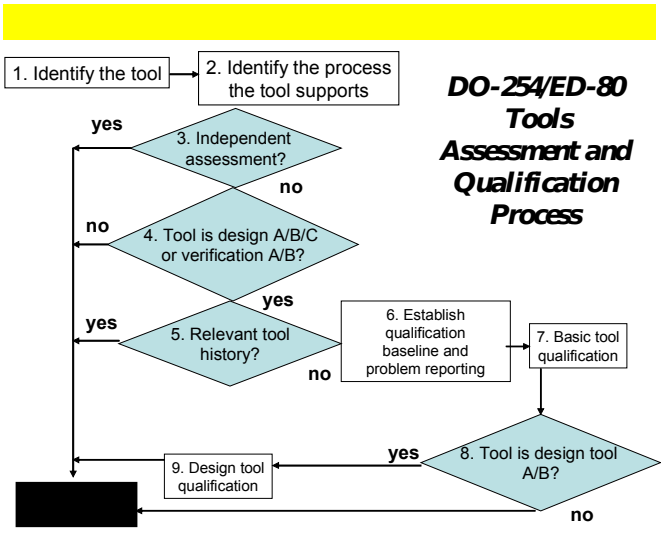

Fig. 2 Tool assessment and qualification process according to DO-254 [2].

leads to the flip-flop output oscillating and not settling to a value within the appropriate delay window. This happens in designs containing multiple asynchronous clocks, when two or more discrete systems communicate. Metastability is a serious problem in safety-critical designs as it causes intermittent failures. A comprehensive verification solution is offered by Mentor Graphics 0-In Clock Domain Crossing (CDC) tool. The tool provides an added assurance that the design will function correctly within the intended system. If one has a specific requirement from the customer or a DER to verify the clock domain crossings and identify and eliminate instances of metastability, then one has to use one of the tool assessment methods. Again, the one suggested is the Independent Output Assessment.

Another verification tool from Mentor Graphics, ModelSim, is discussed by Lange [43] in a view of meeting the DO-254 guidance. The paper outlines the exact ten steps to go through the DO-254 assessment and qualification process, as presented in Figure 2. The suggested way to proceed with tool assessment is to avoid qualification by using an independent output assessment method (Step 3 in Figure 2).

TNI [44] presents Reqtify, tool supporting requirement traceability, impact analysis and automated documentation generation which a ccording to DO-254 classification is a verification tool. Prior to the use of the tool, a tool assessment should be performed to ensure that the tool is capable of performing the particular verification activity to an acceptable level of confidence. The assessment is limited to those functions of the tool used for a specific hardware life cycle activity, not the entire tool.

Dellacherie et al. [45] describe a static formal approach that could be used, in combination with requirements traceability features, to apply formal methods in the design and verification of hardware controllers to support such protocols as ARINC 429, ARINC 629, MIL-STD-1553B, etc. A tool name imPROVE-HDL, a formal property checker, has been used in the design and verification of airborne electronic hardware. Reqtify tool has been used to track the requirements throughout the verification process and to produce coverage reports. According to the authors, using imPROVE-HDL coupled with Reqtify gives confidence that the designers can assure that their bus controllers meet the guidelines outlined in DO-254.

C.Tool Questionnaire

To identify issues and concerns in tool qualification and certification, and help understand the underlying problems, we conducted a survey to collect data on the experiences and opinions concerning the use of programmable logic tools as applied to design and verification of complex electronic hardware according to the RTCA DO-254 guidelines. The objective was to collect feedback, from industry and certification authorities, on assessment and qualification of these tools.

The questionnaire has been developed and distributed during the 2007 National FAA Software & Complex Electronic Hardware Conference, in New Orleans, Louisiana, in July 2007, attended by over 200 participants. In subsequent months, we have also distributed this questionnaire to the participants of two other professional events. It has been made available via DO-254 Users Group website (`http://www.do-254.org/?p=tools`). As a result of these activities a sample of almost forty completely filed responses was received. Even though this may not be a sample fully statistically valid, the collected results make for several interesting observations.

The survey population, by type of the organization, included the majority of respondents from avionics or engine control developers (65%). Over 95% of respondents have technical background (55% bachelor and 45% master degrees) and over 72% have educational background in electronics. While 97% of respondents have more than three years of experience, 59% have more than 12 years. The most frequent respondents' roles relevant to the complex electronics tools include:

- use of the tools for development or verification of systems (62%)
- managing and acting as company's designated engineering representative (26%)
- development of the tools (2%)
- development of components (12%).

The respondents' primary interest was divided between verification (32%), development (27%), hardware (22%) and concept/architecture (18%).

Considering criteria for the selection of tools for use in DO-254 projects (Figure 3), as the most important have been reported the following: the available documentation, ease of qualification, previous tool use, and host platform, followed by the quality of support, tool functionality, tool vendor reputation, and the previous use on airborne project. Selection of a tool for the project is based either on a limited familiarization with the demo version (50%) or an extensive review and test (40%). The approach to review and test the tool by training the personnel and using trial period on a smaller project seems to be prevailing.

For those who have experienced effort to qualify programmable logic tools (only 14% of respondents), the quality of the guidelines is sufficient or appropriate (62%), so is the

ease of finding required information (67%), while the increase of workload was deemed negligible or moderate

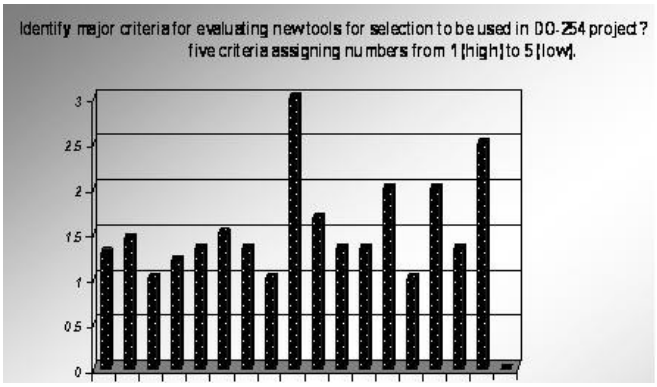

Fig. 3 Tool selection criteria in DO-254 projects (from left to right: vendor reputation, functionality, acquisition cost, compatibility with existing tools, compatibility with development platform, reliability, availability of training, amount of training needed, documentation quality, quality of support, previous familiarity with the tool, performance on internal evaluations, host platform, compatibility with PLDs, previous use on airborne products, tool performance, ease of qualification, other criteria).

(80%). An interesting observation concerns the scale of safety improvement due to qualification: marginal (43%), moderate (21%), noticeable (7%) and significant (29%). Similarly, the question about errors found in the tools may be a source for concern: no errors (11%), few and minor errors (50%), significant and numerous (17%). Despite all this, the satisfaction level towards programmable logic tools was high: more than 96% of respondents marked their satisfaction level as 4 out of 5.

Overall, it is obvious that software tools used in design and verification of complex electronics in safety-critical applications should be scrutinized because of concerns that they may introduce design errors leading to accidents. However, the conducted survey indicated that the most important criteria for tool selection are considered to be: available documentation, ease of qualification, and previous tool use, none of which is technical. In this view, work should be done on developing more objective criteria for tool qualification and conducting experiments with tools to identify their most vulnerable functions that may be a source of subsequent design faults and operational errors. Some of the authors specifically point out that the lack of research investment in certification technologies will have a significant impact on levels of autonomous control approaches that can be properly flight certified, and could lead to limiting capability for future autonomous systems.

V. Summary and Conclusion

The paper makes an attempt to show the role of software certification in development of dependable systems, both from the software and hardware perspective. An important observation is about the increasing role of software tools, which are used to create and verify both software and hardware. An extensive literature review has been presented, focusing on the issues of civil aviation guidance requiring specified level of assurance for the airborne systems, both from the software and hardware perspective.

Both DO-178B and DO-254 guidelines serve industry well and promote rigor and scrutiny required by highly critical systems. However, the relative vagueness of these guidelines causes significant differences in interpretation by industry and should be eliminated. RTCA called a new committee, SC-205, with a charge to revise DO-178B guidance. Possibly, a common ground should be found between RTCA DO-254 and DO-178B guidelines.

References

[1] RTCA DO-178B (EUROCAE ED-12B), *Software Considerations in Airborne Systems and Equipment Certification*, RTCA Inc., Washington, DC, December 1992.

[2] RTCA DO-254 (EUROCAE ED-80), *Design Assurance Guidance for Airborne Electronic Hardware* , RTCA Inc., Washington, DC, April 2000.

[3] Romanski G., Certification of an Operating System as a Reusable Component, *Proc. DASC'02, 21st Digital Avionics Systems Conf.*, Irvine, Calif., October 27-21, 2002, pp. 5.D.3-1/9.

[4] Fachet R., Re-use of Software Components in the IEC-61508 Certification Process, *Proc. IEE COTS & SOUP Seminar*, London, October 21, 2004, pp. 8/1-17.

[5] Parkinson P., L. Kinnan, *Safety-Critical Software Development for Integrated Modular Avionics*, White Paper, Wind River Systems, Alameda, Calif., November 2007.

[6] Moraes R. et al., Component-Based Software Certification Based on Experimental Risk Assessment, *Proc. LADC 2007, 3rd Latin-American Symposium on Dependable Computing*, Morelia, Mexico, September 26-28, 2007, pp. 179-197.

[7] Maxey B., COTS Integration in Safety Critical Systems Using RTCA/DO-178B Guidelines, Proc. ICCBSS 2003, 2nd International Conference on COTS-Based Software Systems, Ottawa, Ont., February 10-13, 2003, pp. 134-142.

[8] Halang W., J. Zalewski, Programming Languages for Use In Safety Related Applications, *Annual Reviews in Control*, Vol. 27, pp. 39-45, 2003.

[9] Goodenough J.B., The Ada Compiler Validation Capability, *ACM SIGPLAN Notices* , Vol. 15 , No. 11, pp. 1-8, November 1980.

[10] Santhanam V., The Anatomy of an FAA-Qualifiable Ada Subset Compiler, *Ada Letters*, Vol. 23, No. 1, March 2003, pp. 40-43 (Proc. SIGAda'02, Houston, Texas, December 8-12, 2002).

[11] Comar C., R. Dewar, G. Dismukes, Certification & Object Orientation: The New Ada Answer, *Proc. ERTS 2006, 3rd Embedded Real-Time Systems Conference* , Toulouse, France, January 25-27, 2006.

[12] Brosgol B.M., Ada 2005: A Language for High-Integrity Applications, *CrossTalk – The Journal of Defense Systems* , Vol. 19, No. 8, pp. 8-11, August 2006.

[13] Amey P., R. Chapman, N. White, Smart Certification of Mixed Criticality Systems, *Proc. Ada-Europe 2005, 10th Intern. Conf. on Reliable Software Technologies*, York, UK, June 20-24, 2005, pp. 144-155.

[14] Hatton L., Safer Language Subsets: An Overview and Case History - MISRA C, *Information and Software Technology*, Vol. 46, No. 7, pp. 465-472, 2004.

[15] Subbiah S., S. Nagaraj, Issues with Object Orientation in Verifying Safety-Critical Systems, *Proc. ISORC'03, 6th International IEEE Symposium on Object-Oriented Real-Time Distributed Computing* , Hakodate, Hokkaido, Japan, May 14-16, 2003.

[16] Kornecki A., N. Brixius, J. Zalewski, *Assessment of Software Development Tools for Safety-Critical Real-Time Systems*, Technical Report DOT/FAA/AR-06/36, Federal Aviation Administration, Washington, DC, July 2007.

[17] Kornecki A., J. Zalewski, Experimental Evaluation of Software Development Tools for Safety-Critical Real-Time Systems, *Innovations in Systems and Software Engineering – A NASA Journal*, Vol. 1, No. 2, pp. 176-188, September 2005.

[18] Kornecki A., J. Zalewski, The Qualification of Software Development Tools from the DO-178B Certification Perspective, *Crosstalk - The Journal of Defense Software Engineering*, Vol. 19 , No. 4, pp. 19-23, April 2006.

[19] Santhanam V. et al, *Software Verification Tools Assessment Study*, Technical Report DOT/FAA/AR-06/54, Federal Aviation Administration, Washington, DC, June 2007.
[20] Dewar R., B. Brosgol, Using Static Analysis Tools for Safety Certification, *VMEbus Systems* , pp. 28-30, April 2006.
[21] Dewar R.B.K., Safety Critical Design for Secure Systems, *EE Times-India*, July 2006.
[22] Santhanam U., Automating Software Module Testing for FAA Certification, *Ada Letters*, Vol. 21, No. 4, pp. 31-37, December 2001 (Proc. SIGAda'01, Bloomington, MN, Sept. 30 – Oct. 4, 2001).
[23] Souyris J., D. Delmas, Exterimental Assessment of Astreé on Safety-Critical Avionics Software, *Proc. SAFECOMP 2007, 26th Intern. Conf. on Computer Safety, Reliability and Security*, Nuremberg, Germany, Sept. 18-21, 2007.
[24] *DO-178B and McCabe IQ*, McCabe Software, Warwick, RI, December 2006.
[25] *Safety Critical Systems Club Tools Directory*, London, UK, http://www.scsc.org.uk/tools.html
[26] Hilderman V., T. Baghai, Avionics Hardware Must Now Meet Same FAA Requirements as Airborne Software, *COTS Journal*, Vol. 5, No. 9, pp. 32-36, September 2003.
[27] Cole P., M. Beeby, Safe COTS Graphics Solutions: Impact of DO-254 on the Use of COTS Graphics Devices for Avionics, *Proc. DASC'04, 23rd Digital Avionics Systems Conference*, Salt Lake City, Utah, October 24-28, 2004, pp. 8A2-8.1/7.
[28] Federal Aviation Administration, *Advisory Circular AC 20-152, RTCA Document RTCA/DO-254 Design Assurance Guidance for Airborne Electronic Hardware*, June 30, 2005.
[29] Miner P.S. et al., A Case-Study Application of RTCA DO-254: Design Assurance Guidance for Airborne Electronic Hardware, *Proc. DASC 2000, 19th Digital Avionics Systems Conference*, Philadelphia, PA, October 7-13, 2000, Vol. 1, pp. 1A1/1 – 1A1/8.
[30] Baghai T., L. Burgaud, *DO254 Package Process and Checklists: Overview & Compliance with RTCA/DO-254 Document*, White Paper, DO-254 Users Group, March 2004.
[31] Glazebrook I., *The Certification of Complex Hardware Programmable Logic Devices (PLDs) for Military Applications*, White Paper, DNV UK, London, 2007.
[32] Pampagnin P., J.F. Menis, *DO254-ED80 for High Performance and High Reliable Electronic Components*, Internal Paper, Barco-Siles S.A., Peynier, France, 2007.
[33] Karlsson K., H. Forsberg, Emerging Verification Methods for Complex Hardware in Avionics, *Proc. DASC '05, 24th Digital Avionics Systems Conference*, Washington, DC, Oct.-30-Nov. 3, 2005, Vol. 1, pp. 6.B.1-1/11.
[34] Lange M., T.J. Boer, *Effective Functional Verification Methodologies for DO-254 Level A/B and Other Safety-Critical Devices*, White Paper, Rev. 1.1, Mentor Graphics, Wilsonville, Ore., 2007.
[35] Keithan J.P. et al., The Use of Advanced Verification Methods to Address DO-254 Design Assurance, *Proc. 2008 IEEE Aerospace Conference*, Big Sky, Montana, March 1-8, 2008.
[36] Lange M., T. Dewey, Achieving Quality and Traceability in FPGA/ASIC Flow for DO-254 Aviation Projects, *Proc. 2008 IEEE Aerospace Conference* , Big Sky, Montana, March 1-8, 2008.
[37] Lee M., T. Dewey, Accelerating DO-254 for ASIC/FPGA Designs, *VME and Critical Systems*, pp. 28-30, June 2007.
[38] Sysenko I., R. Pragasam, Hardware-based Solution Aides: Design Assurance for Airborne Systems, *Military Embedded Systems*, pp. 26-28, July 2007.
[39] Land I., I. Bryant, FPGA IP Verification for Use in Severe Environments, *Proc. 2005 Annual MAPLD International Conference*, Washington, DC, Sept. 7-9, 2005.
[40] Lundquist P., *Certification of Actel Fusion according to RTCA DO-254*. Master Thesis, Report LiTH-ISY-EX-ET-07/0332-SE, Linköping University, Sweden, May 4, 2007.
[41] Aldec Corp., *DO-254 Hardware Verification: Prototyping with Vectors Mode*. White Paper, Rev. 1.2, Henderson, Nevada, June 2007.
[42] Lange M., Automated CDC Verification Protects Complex Electronic Hardware from Metastability Issues, *VME Critical Systems*, Vol. 26, No. 3, pp. 24-26, August 2008.
[43] Lange M., *Assessing the ModelSim Tool for Use in DO-254 and ED-80 Projects*, White Paper, Mentor Graphics Corp., Wilsonville, Ore., May 2007.
[44] Baghai T., L. Burgaud, *Reqtify: Product Compliance with RTCA/DO-254 Document*, TNI-Valiosys, Caen, France, May 2006
[45] Dellacherie S., L. Burgaud, P. di Crescenzo, Improve – HDL: A DO-254 Formal Property Checker Used for Design and Verification of Avionics Protocol Controllers, *Proc. DASC'03, 22nd Digital Avionics Systems Conf.*, Indianapolis, Oct. 12-16, 2003, Vol. 1, pp. 1.A.1-1.1-8.