

# Software Certification: Methods and Tools

Edited by

Darren Cofer<sup>1</sup>, John Hatcliff<sup>2</sup>, Michaela Huhn<sup>3</sup>, and  
Mark Lawford<sup>4</sup>

1 Rockwell Collins – Cedar Rapids, US, [ddcofer@rockwellcollins.com](mailto:ddcofer@rockwellcollins.com)

2 Kansas State University, US, [hatcliff@cis.ksu.edu](mailto:hatcliff@cis.ksu.edu)

3 TU Clausthal, DE, [michaela.huhn@tu-clausthal.de](mailto:michaela.huhn@tu-clausthal.de)

4 McMaster University – Hamilton, CA, [lawford@McMaster.CA](mailto:lawford@McMaster.CA)

---

## Abstract

---

With the pervasive deployment of software in dependable systems used in everyday life, society is increasingly demanding that software used in critical systems must meet minimum safety, security and reliability standards. *Certification* is the procedure by which an authorized person or agency assesses and verifies characteristics of a system or product in accordance with established requirements, standards, or regulations. For software, it encompasses traditional notions of verification, but also includes the evidence, tools, methods, and personnel qualifications that are needed to convince the certification authority that the system or product conforms to the relevant standard. Manufacturers of these systems need consistent and effective guidelines as to what constitutes acceptable evidence of software quality, and how to achieve it.

Compared to process-oriented certification procedures, recent approaches provide evidence for dependability by the thorough evaluation of the product itself and the adequacy, coverage and maturity of design and quality assurance methods. Substantial progress has been made in areas including safety and assurance cases, the conceptual foundation of evidence and formal methods, and tooling for software design and verification. New approaches are necessary to develop holistic and cost-effective methodologies and to provide integrated tool support for creating certifiable software-intensive systems, as well as product-focused approaches to certifying these systems.

Experts from academia and industrial practitioners met in the Dagstuhl Seminar 13051 “Software Certification: Methods and Tools” to discuss and software certification challenges, best practices, and the latest advances in certification technologies in several different software-intensive domains (automotive, aircraft, medical, nuclear, and rail).

**Seminar** 27. January to 01. February, 2013 – [www.dagstuhl.de/13051](http://www.dagstuhl.de/13051)

**1998 ACM Subject Classification** D.2.0 Software Engineering / General, D.2.4 Software/Program Verification, D.2.9 Management / Software Quality Assurance, I.6.4 Model Validation and Analysis, K.4.1 Public Policy Issues / Human Safety, K.5.2 Governmental Issues / regulation, K.6.3 Software Management / Software Process

**Keywords and phrases** dependable systems, safety, security, certification, formal methods, model-driven development, validation & verification, tools

**Digital Object Identifier** 10.4230/DagRep.3.1.111

**Edited in cooperation with** Sara Bessling



Except where otherwise noted, content of this report is licensed under a Creative Commons BY 3.0 Unported license

Software Certification: Methods and Tools, *Dagstuhl Reports*, Vol. 3, Issue 1, pp. 111–148

Editors: Darren Cofer, John Hatcliff, Michaela Huhn, and Mark Lawford



DAGSTUHL  
REPORTS

Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

## 1 Executive Summary

*Darren Cofer*

*John Hatcliff*

*Michaela Huhn*

*Mark Lawford*

License © Creative Commons BY 3.0 Unported license  
© Darren Cofer, John Hatcliff, Michaela Huhn, and Mark Lawford

### Context

An increasingly important requirement for success in many domains is the ability to cost-effectively develop and certify software for critical systems (e.g. pacemakers, health monitoring equipment, core banking applications, financial reporting, nuclear reactors, rail automation and active safety in vehicles etc.). Software errors in each of these domains continue to lead to catastrophic system failures, sometimes resulting in loss of life. A recent report by the U.S. National Academy of Sciences [1], concludes that “new techniques and methods will be required in order to build future software systems to the level of dependability that will be required...In the future, more pervasive deployment of software...could lead to more catastrophic failures unless improvements are made.” Thus, society is increasingly demanding that software used in critical systems must meet minimum safety, security and reliability standards. Manufacturers of these systems are in the unenviable position of not having consistent and effective guidelines as to what constitutes acceptable evidence of software quality, and how to achieve it. This drives up the cost of producing these systems without producing a commensurate improvement in dependability.

Multiple trends and activities (a) point to the changing nature of development of certified systems and (b) indicate the need for community-wide efforts to assess and form a vision of the future for development of certified systems.

### New and Evolving Standards

To adapt to the significant changes in the role of software in dependable systems and to improve current industrial practice in software engineering, international standards like the IEC 61508 are currently under revision. DO-178C governing certification of software in commercial aircraft has recently been revised to accommodate the use of software technologies such as formal methods and model-based development processes. In several other software-intensive domains new domain-specific standards are being developed.

### Process- vs. Product-oriented Certification

In practice, current certification of software-intensive systems is primarily process based. A reliance on process oriented standards has established a certification practice that is dominated by assessing process-related documents and marking off checklists that are derived from the recommendation annexes of the standards or so-called “approved practice in use”. Thorough evaluation of the product itself or the adequacy, coverage and maturity of design and quality assurance methods are sometimes neglected because there is currently no fundamental agreement on software engineering principles and product qualities to achieve demonstrably dependable software. An alternative to process-oriented certification regimes is “safety and assurance cases” [7]. In Europe, and particularly in the UK, assurance cases have been adopted as a product centric alternative approach to certification and are widely used in

practice already. Recently the U.S. FDA has issued guidance documents recommending the use of assurance cases in submissions for approval of infusion pumps. However, while assurance cases offer some product oriented focus to certification, the lack of standardization of safety and assurance case arguments has its own pitfalls [8].

### **Advances in Formal Methods**

In academia, research on formal methods has made substantial progress with respect to scalability and coverage recently, e.g. in tool-supported model-based design and code generation, but also in the area of software model checking or timing analysis [2]. Thus, formally assuring safety requirements has become feasible at least on the level of components. Nevertheless, research usually focuses on specific techniques, thereby often neglecting the cross-cutting nature of dependability and the need of providing traceable evidence.

### **Software Development Trends**

Two trends relevant in industrial software development for critical systems are the success of model-based design environments that support automated code generation and the need to integrate pre-developed or Commercial Off The Shelf (COTS) software components: (1) Model based tools facilitate rapid prototyping and validation and verification in earlier design phases than traditional processes, but with a price of higher effort in the design phases performed by well-trained and experienced personnel. Software quality will only benefit from these approaches, if certification procedures are adapted towards a cost-effective assessment on the level of models wherever it is adequate. For instance, if model based tools are supported by V & V tools that perform some verification at design time, how does this affect certification standards that require independent design and V & V teams? (2) Evidence based upon prior usage and operating history are typically key components in making decisions in industry about the “fitness for use” of a pre-developed software application or component. However, platform-specific and environmental constraints on the usage are sometimes not specified in detail which has lead to catastrophic failures in the past.

### **Community-building Activities**

Various community-building organizations are being formed drive research, education, and cross-domain coordination in the area of software certification. For instance, the Software Certification Consortium (SCC) was formed in 2007 as a North American initiative to promote product based software certification. Its members are drawn from regulators, industry and academia. SCC has been successful in highlighting shortcomings in current certification regimes and in providing challenge problems and example certification artifacts to the broader community.

### **Seminar Topics and Goals**

The Dagstuhl Seminar 13051 *Software Certification: Methods and Tools* brought together experts for the purpose of assessing the current state of practice, identifying challenges, promising techniques/methods, and for creating a road map for future research, education, and standards development in the area of certification of software and systems.

The seminar addressed the following topics:

- Identification of the challenges, regulatory bodies, primary certification standards, typical development and certification processes in variety of safety-critical domains including avionics, automotive, medical systems, and rail, as well as cross-cutting aspects of security certification.
- Developing a *rational basis* for the primary activities in certification. This included work on the interrelation between i) how we develop software in a way that facilitates certification; and, ii) how we collect and use evidence about software products to evaluate whether they should or should not be certified for use, and iii) cost-benefits issues in certification.
- Pros and cons of assurance-cases in regulatory regimes, assessing the confidence given by assurance cases, new techniques for presenting assurance case arguments, tools for managing the collection of evidence and organization of arguments for assurance cases, and the relationship between assurance cases and software certification standards such as DO-178C.
- The use of tools and open source infrastructure in certification, along with new approaches and guidelines for qualifying tools for use in development of certified systems.
- The latest advances in relevant formal methods for software verification, and integrating formal method with other quality assurance techniques such as testing in the context of certified system development.
- The increasing use of “systems of systems” in safety-critical domains, and the need for new approaches supporting compositional certification and reuse of components in the context of certified systems.
- The structure, nature, use, of current certification standards, current business models and organizational principles for developing standards, and how these aspects might be evolved to better address the needs of the community.
- Strategies for managing the complexity of software intensive systems, including model based development, refinement-based methodologies, and generative techniques.
- Challenges problems, infrastructure, and pedagogical resources to support research and education for both academia and industry in the area of certified system development.

### Seminar Participants and Activities

41 researchers participated in the “Software Certification” Seminar, 21 academic researchers, 10 are affiliated to research institutes and 10 experts from industries proving the strategic relevance of the subject to both, research and practice. With about 40% the portion of North American participants was remarkable high.

The seminar started with an introductory session on Monday morning at which the organizers recapitulated the outline, the objectives, and goals of the seminar. Each participant shortly introduced him/herself, his/her scientific background and personal goals for the seminar week. Then senior experts gave an overview on software certification in different domains, namely the avionic, nuclear, medical devices, automotive, and the rail domain. Monday afternoon ended with a discussion on the major differences and similarities between software assessment in the domains and cross-domain challenges. From Tuesday to Thursday experts presented their work. Panel discussions, challenge problem advertisements as well as working group sessions took place in the afternoons and evenings. A wide range of topics was covered including assurance cases and the fundamentals of how to achieve evidence, tool support to software assessment in the certification process, experience reports and new

methodologies for the medical device domain, model based design methods appropriate to certification, issues in cloud security and security certification, tools and methods for static analysis, formal verification and testing. On Thursday evening we had a fruitful discussion with the participants of the Dagstuhl seminar on “Multicore Enablement for Embedded and Cyber-Physical Systems” organized by Andreas Herkersdorf, Michael Hinchey, and Michael Paulitsch that was held in parallel. Among others the following questions were discussed: What are the requests on predictability that have to be satisfied by multicore architectures to be well suited for dependable systems? What are the compelling cyberphysical dependable applications that need multicore architectures? What mechanisms known from dependable software development may be transferred to multicore architecture design and vice versa? Friday was dedicated to working groups as well as outlining and scheduling post seminar proceedings in which we plan to summarize the state of the art in software certification and the results of the seminar. The areas identified by the plenum to be most relevant for further progress on software certification are:

- Fundamentals on confidence and evidence
- Compositional certification
- Education on dependable systems and certification
- Tool qualification
- Security
- Methods for the development of certifiable software and methods supporting certification

#### References

- 1 D. Jackson, M. Thomas, L. Millett. “Software for Dependable Systems: Sufficient Evidence?” Committee on Certifiably Dependable Software Systems, National Research Council, National Academies Press, 2007.
- 2 M. Huhn, H. Hungar. UML for software safety and certification – Model-based development of safety-critical software-intensive systems. In H. Giese, G. Karsai, E. Lee, B. Rumpe, and B. Schätz (Eds.): Model-Based Engineering of Embedded Real-Time Systems – Int’l Dagstuhl Workshop, Dagstuhl Castle, Germany, November 4–9, 2007. Revised Selected Papers, LNCS 6100, Springer, pp. 203–240. 2011. DOI: 10.1007/978-3-642-16277-0\_8
- 3 M. Huhn, A. Zechner. Analysing Dependability Case Arguments Using Quality Models. In B. Buth, G. Rabe, and T. Seyfarth (Eds.): 28th Int’l. Conf. on Computer Safety, Reliability, and Security (SAFECOMP), LNCS 5775, Springer, pp. 118–131, 2009. DOI: 10.1007/978-3-642-04468-7\_11
- 4 A. Wassying, T.S.E. Maibaum, M. Lawford, On Software Certification: We Need Product-Focused Approaches. C. Choppy and O. Sokolsky (Eds.): Monterey Workshop 2008, LNCS Vol. 6028, Springer, pp. 250–274, 2010. DOI: 10.1007/978-3-642-12566-9\_13
- 5 J. Hatcliff, M.P.E. Heimdahl, M. Lawford, T.S.E. Maibaum, A. Wassying, F.L. Wurden. A Software Certification Consortium and its Top 9 Hurdles. In Proc.of the First Workshop on Certification of Safety-Critical Software Controlled Systems (SafeCert 2008), ENTCS, Vol. 238, No. 4, pp. 11–17, 2009. DOI: 10.1016/j.entcs.2009.09.002
- 6 FDA, “FDA Launches Initiative to Reduce Infusion Pump Risks,” News Release, April 23, 2010 (see: <http://www.fda.gov/NewsEvents/Newsroom/PressAnnouncements/ucm209042.htm>)
- 7 R. Bloomfield and P. Bishop. Safety and assurance cases: Past, present and possible future – an Adelard perspective. In C. Dale, T. Anderson (Eds.): Making Systems Safer, Proc. of the Eighteenth Safety-Critical Systems Symp., Bristol, UK (February 2010), pp. 51–67.
- 8 A. Wassying, T.S.E. Maibaum, M. Lawford and H. Behr. Is there a case against safety cases? Submitted to post-proceedings volume of Monterey 2010 Workshop, to be published in LNCS.

## 2 Table of Contents

### Executive Summary

*Darren Cofer, John Hatcliff, Michaela Huhn, and Mark Lawford* . . . . . 112

### Overview of Talks

Modeling Requirements for Embedded Systems with RDAL <i>Dominique Blowin</i> . . . . .	119
Technology Infusion Study for DO-333 <i>Darren Cofer</i> . . . . .	119
Integrating Formal Program Verification with Testing <i>Cyrille Comar</i> . . . . .	120
Functional Safety and Certification of Automotive E/E systems <i>Mirko Conrad</i> . . . . .	120
Abstraction, Fidelity and (In-)Competence: modelling cyber-physical systems and systems of systems <i>John S. Fitzgerald</i> . . . . .	121
What is Mission-Assurance? <i>Kim R. Fowler</i> . . . . .	121
A naive look at software certification practices – and proposals for enhancement <i>Hubert Garavel</i> . . . . .	122
Bringing evidence-based arguments into practice <i>Janusz Gorski</i> . . . . .	122
Static Analysis of Real-Time Embedded Systems with REK <i>Arie Gurfinkel</i> . . . . .	123
Certification for Medical Devices and Systems: An Overview and Challenges <i>John Hatcliff</i> . . . . .	124
Requirements Specification and Supporting Artifacts for an Open Source Patient-Controlled Analgesic Pump <i>John Hatcliff</i> . . . . .	124
Concerning the implicit DO-178C assurance case <i>Michael Holloway</i> . . . . .	125
Software verification in the medical domain <i>Jozef Hooman</i> . . . . .	125
Bridging the modeling/verification gap <i>Jerôme Hugues</i> . . . . .	125
Opening up the Verification and Validation of Safety-Critical Software <i>Hardi Hungar</i> . . . . .	126
Using Code Analysis Tools for Software Safety Certification <i>Daniel Kaestner</i> . . . . .	126
Towards an Effective Safety Demonstration Framework <i>Peter Karpati</i> . . . . .	127

Software Certification: Where is Confidence Won and Lost? <i>Tim Kelly</i> . . . . .	127
User Assembled Medical System of Systems <i>Andrew King</i> . . . . .	128
Three Challenges <i>John C. Knight</i> . . . . .	128
Certification of Medical Device Composition <i>Brian Larson</i> . . . . .	128
Bayesian Probabilistic Approaches to Confidence are Impossible: The Need for a Baconian Approach (pace Jonathan Cohen) <i>Tom S. Maibaum</i> . . . . .	129
Software Certification: The Return on Investment? <i>John McDermid</i> . . . . .	129
Refinement may help for Certification <i>Dominique Mery</i> . . . . .	130
Certification Challenges for Software With Uncertainty <i>Richard F. Paige</i> . . . . .	131
Models and Certification <i>Andras Pataricza</i> . . . . .	132
From Tool Qualification to Tool Chain Design <i>Jan Philipps</i> . . . . .	132
Cloud Security: Information Segregation and Data Privacy <i>Julia Rubin</i> . . . . .	133
Logic and Epistemology in Assurance Cases <i>John Rushby</i> . . . . .	133
Model-Based Development and Functional Safety <i>Bernhard Schaetz</i> . . . . .	134
Software Certification Challenges in the Nuclear Power Domain <i>Alan Wassynq</i> . . . . .	134
Certification of Medical Information Systems – A paradigm shift: from devices to systems, from functions to data <i>Jens H. Weber</i> . . . . .	134
Software certification in aeronautics <i>Virginie Wiels</i> . . . . .	135
Some experience and remarks on security certification at industry <i>David von Oheimb</i> . . . . .	135
<b>Overview of Working Groups</b>	
Challenges: Compositional Certification . . . . .	135
Challenges: Education and Challenge Problems . . . . .	139
Challenges: Security . . . . .	141
Challenges: Tool Qualification . . . . .	142

**118      13051 – Software Certification: Methods and Tools**

Intellectual Basis for Certification & Confidence . . . . . 144  
Methods for Developing Certifiable Systems and Methods of Certifying Systems . . 146  
**Participants . . . . . 148**



## 3 Overview of Talks

### 3.1 Modeling Requirements for Embedded Systems with RDAL

*Dominique Blouin (Université de Bretagne Sud, FR)*

**License** © Creative Commons BY 3.0 Unported license  
© Dominique Blouin

**Joint work of** Blouin, Dominique; Turki, Skander, Senn Eric

**Main reference** D. Blouin, E. Senn, S. Turki, “Defining an annex language to the architecture analysis and design language for requirements engineering activities support,” in Proc. of Model-Driven Requirements Engineering Workshop (MoDRE’11), pp. 11–20, IEEE, 2011.

**URL** <http://dx.doi.org/10.1109/MoDRE.2011.6045362>

In this talk, I will introduce the Requirements Definition and Analysis Language (RDAL) that we are developing as an annex of the Architecture Analysis and Design Language (AADL). I will present the needs for such language, its main features and show how it can be used to formalize requirements specifications supporting requirements engineering best practices such as those of the FAA Requirements Engineering Management Handbook. I will also present the modeling of a concrete example from the FAA handbook with RDAL, and discuss some modeling issues and inconsistencies of the natural language specification revealed by the process of formalizing the specification. I would like to get inputs on the potential impact of finding these inconsistencies on the system development process and the benefits of the modeling activity taking into account the overhead it implies.

### 3.2 Technology Infusion Study for DO-333

*Darren Cofer (Rockwell Collins – Minneapolis MN, US)*

**License** © Creative Commons BY 3.0 Unported license  
© Darren Cofer

**Joint work of** Cofer, Darren; Miller, Steven

In 2012, RTCA published DO-178C, DO-278A, and DO-333, which together define a framework for applying formal methods in the certification of airborne and air traffic management systems. However, there remain significant challenges to the successful infusion of formal methods into development and certification workflows in the aviation industry. Under NASA funding, Boeing and Rockwell Collins are developing technical material that will help industry to effectively apply formal methods in a DO-178C/DO-333 context. Our work includes a survey of currently available formal methods and tools that are most relevant for verification and certification of airborne software. We have also developed an integrated case study to demonstrate the use of some of these tools to satisfy DO-178C certification objectives using the augmented guidance in DO-333. The case study is based on a fault-tolerant flight control system that includes requirements and design artifacts specified using PVS, Simulink/Stateflow, and source code. We have verified different aspects of the system design using theorem proving, model checking, and abstract interpretation, and show how various certification objectives are satisfied using these formal techniques. The survey and the case study will form the basis for a new formal methods guidebook, which will be publicly available along with all of the models and verification artifacts. Both the technical descriptions and the examples will be presented at varying levels of detail, suitable for system developers, certifiers, and other stakeholders.

### 3.3 Integrating Formal Program Verification with Testing

*Cyrille Comar (AdaCore, Paris, FR)*

License  Creative Commons BY 3.0 Unported license  
© Cyrille Comar

Joint work of Comar, Cyrille; Kanig, Johannes; Moy, Yannick

The Hilite project proposes a framework that offers the possibility of using formal and non-formal verification of requirements expressed in the form of subprogram contracts in a DO-178C context. In particular, we explore the conditions for validating an approach mixing formal verification and testing whose goal is to reduce the costs of testing and ease the adoption of formal verification in the industry.

### 3.4 Functional Safety and Certification of Automotive E/E systems

*Mirko Conrad (The MathWorks GmbH – Ismaning, DE)*

License  Creative Commons BY 3.0 Unported license  
© Mirko Conrad


ISO 26262 “Road vehicles – Functional safety” is a set of safety standards for electrical and/or electronic (E/E) systems installed in series production passenger cars. ISO 26262 constitutes the adaptation of IEC 61508 to comply with needs specific to the application sector of E/E systems within such road vehicles. ISO 26262 which was published in 2011 provides:

- An automotive safety lifecycle incl. means for tailoring the necessary activities
- An automotive-specific risk-based approach to determine integrity levels (Automotive Safety Integrity Levels, ASILs) and uses ASILs to specify applicable requirements to avoid unreasonable residual risk
- Requirements for validation and confirmation measures to ensure a sufficient and acceptable level of safety being achieved
- Requirements for OEM-supplier relationship.

Functional safety as per the standard is defined as “absence of unreasonable risk due to hazards caused by malfunctioning behaviour of E/E systems”. ISO 26262 does not have a notion of certification, but it sets out three types of so called confirmation measures (confirmation review, functional safety audit, and functional safety assessment). The standard also provides a process to establish the required confidence in the correct functioning of software tools. This process comprises two steps, tool classification and subsequent tool qualification (if applicable). The talk provides an overview of key concepts of ISO 26262 that are related to the topic area of the seminar.

### 3.5 Abstraction, Fidelity and (In-)Competence: modelling cyber-physical systems and systems of systems

*John S. Fitzgerald (Newcastle University, GB)*

License  Creative Commons BY 3.0 Unported license  
© John S. Fitzgerald

Joint work of Fitzgerald, John S.; Larsen, Peter G; Verhoef, Marcel HG; Pierce, K; Gamble, C

Formal model-based methods have been advocated as a means of gaining assurance in the early stages of product development. Formal notations are said to promote careful abstraction, the explicit recording of assumptions, and the elimination of infeasible designs. The range of applications of formal methods, and the capability of analysis techniques, using proof and model-checking for example, have grown in the last quarter century. However, the use of a notation that happens to have formal semantics is not in itself a basis for making substantive judgements about any product that may ultimately be derived from the model. What, then, are the costs, benefits and risks of formal modelling and analysis?

My talk focuses on the new and challenging domains of cyber-physical systems in which there is a close interaction between networked computing devices and the physical environment, and in systems of systems, which are composed of independently owned and managed constituents about which we have limited knowledge and over which we have limited control. What can formal techniques contribute to the assurance of such systems?

I use examples drawn from recent work on the following projects in which formal model-based techniques are developed with the aim of help to de-risk development by eliminating inadequate or infeasible designs at early stages. In both cases, the quality of models depends on balancing abstraction and fidelity while not compromising competence.

- DESTECs (Design Support and Tooling for Embedded Control Software) [www.destecs.org](http://www.destecs.org) defines methods and tools for collaborative modelling and co-simulation for embedded systems, linking discrete-event models of software to continuous-time models of controlled plant and environment. A reconciled operational semantics for simulators in both domains enables systematic exploration of design spaces.
- COMPASS (Comprehensive Modelling for Advanced Systems of Systems) [www.compass-research.eu](http://www.compass-research.eu) is developing formal modelling techniques for Systems of Systems. This entails providing common semantic bases for a range of aspects including data, functionality, architectural structure, time and mobility.

### 3.6 What is Mission-Assurance?

*Kim R. Fowler (Kansas State University, US)*


License  Creative Commons BY 3.0 Unported license  
© Kim R. Fowler

I will present several case studies, from military, medical, and appliance industries, as examples of dependable systems, which illustrate some aspects of mission-assurance. We, the seminar participants, will discuss what “dependable” and “mission-assurance” really mean in the context of each of these case studies.

I will present and discuss with you some components of development processes that lead towards mission-assurance in systems. My primary conclusion is: Appropriate processes and procedures in design and development lead to dependable systems and mission-assurance, not blind application of all possible techniques.

### 3.7 A naive look at software certification practices – and proposals for enhancement

*Hubert Garavel (INRIA Rhône-Alpes, FR)*

License  Creative Commons BY 3.0 Unported license  
© Hubert Garavel

In this talk I will provide the feedback of an academic computer scientist confronted to the current multiplicity of software certification standards. The current situation seems far from optimal because of the diversity of vocabulary and concepts, and because key ideas of software engineering and formal methods seem to be missing from current standards. Based on these remarks, some proposals for enhancing certification practices are formulated.

### 3.8 Bringing evidence-based arguments into practice

*Janusz Gorski (Gdansk University of Technology, PL)*

License  Creative Commons BY 3.0 Unported license  
© Janusz Gorski

Evidence-based arguments have a potential to strengthen trust relationships in different contexts. To support their wider application, several problems have to be solved, including: choosing an adequate argument model, integration of arguments and the supporting evidence, diversity of evidence formats (e.g. text, graphics, video stream, audio), diversity of the evidence repositories, user-friendly interface, argument assessment (including multiple assessments and diverse assessment mechanisms), communication of the argument assessment results, deployment models for supporting tools, information security (in particular, security of an argument and security of the evidence supporting the argument), scalability, version management and so on.

To address this and the related problems we are developing the TRUST-IT methodology [2] and the related platform of software services, called NOR-STA [1]. TRUST-IT is focused on representation and assessment of evidence based arguments and on their possible usage scenarios [10, 8, 7, 5, 6]. The arguments are represented graphically, with the help of NOR-STA software services which are deployed in accordance with the SaaS (Software-As-A-Service) cloud computing model. The “strength” of an argument can be appraised by an independent assessor using the appraisal mechanism based on Dempster-Shafer theory. The use of other, application domain specific argument appraisal mechanisms is also supported. The results of argument assessment are visualized by coloring the argumentation tree which provides for effective and efficient communication and decision making support.

TRUST-IT and NOR-STA have been already used in different application scenarios, including justification of safety, security and privacy properties of IT systems and services developed in four different European research projects (DRIVE, PIPS, ANGEL and DECOS) [4], justification of the trustworthiness of the assessment criteria of web-based sources of medical information (applied by the Health-On-the Net foundation based in Switzerland) [3], and are presently used to support the processes of achieving and assessing standards conformance in different domains, including accreditation standards for hospitals [9], a standard for risk management in outsourcing processes, CAF (Common Assessment Framework) – a set of guidelines for self-assessment and self-improvement of public organizations, and presently we begin to support HACCP (Hazard Analysis and Critical Control Points), a systematic

preventive approach to food safety and allergenic, chemical, and biological hazards. New application scenarios, including parallel monitoring of critical requirements in different sites and automatic assessment of selected claims are under investigation.

## References

- 1 <http://www.nor-sta.eu/en>
- 2 [http://iag.pg.gda.pl/iag/?s=research&p=trust\\_cases](http://iag.pg.gda.pl/iag/?s=research&p=trust_cases)
- 3 <http://www.hon.ch/Global/copyright.html>
- 4 Górski J, Jarzêbowicz A, Miler J, Witkowicz M, Czyznikiewicz J, Jar P: Supporting Assurance by Evidence-Based Argument Services, SAFECOMP Workshops 2012, Springer-Verlag Berlin, Heidelberg, pp. 417–426
- 5 Cyra Ł., Górski J., Support for argument structures review and assessment, Reliability Engineering and System Safety, Elsevier, Volume 96, 2011, pp. 26–37
- 6 Cyra Ł., Górski J., SCF – a Framework Supporting Achieving and Assessing Conformity with Standards, Computer Standards & Interfaces, Elsevier, Volume 33 Issue 1, January, 2011, pp. 80–95
- 7 Górski J., Jarzêbowicz A., Leszczyna R., Miler J., Olszewski M.: Trust case: justifying trust in IT solution, Proc. Safecomp Conference, Reliability Engineering and System Safety, Elsevier, vol. 89/1, 2005, pp. 33–47. 8
- 8 Górski J.: Trust-IT – a framework for trust cases, Workshop on Assurance Cases for Security – The Metrics Challenge, Proc. of DSN 2007, June 25-28, Edinburgh, UK, 2007, pp. 204–209.
- 9 Górski J., Jarzêbowicz A., Miler J., Validation Of Services Supporting Healthcare Standards Conformance, Journal on Metrology and Measurement Systems, vol. XIX, No. 2, 2012, pp. 269–284
- 10 Górski J., Trust Case – a case for trustworthiness of IT infrastructures, in Cyberspace Security and Defense: Research Issues, NATO Science Series II: Mathematics, Physics and Chemistry, Vol. 196, Springer-Verlag, 2005, pp. 125–142

## 3.9 Static Analysis of Real-Time Embedded Systems with REK

*Arie Gurfinkel (CMU – Pittsburgh PA, US)*

**License** © Creative Commons BY 3.0 Unported license  
© Arie Gurfinkel

**Joint work of** Gurfinkel, Arie; Chaki, Sagar; Strichman, Ofer; Kong, Soonho

**Main reference** S. Chaki, A. Gurfinkel, S. Kong, O. Strichman, “Compositional Sequentialization of Periodic Programs,” in Proc. of the 14 Int’l Conf. on Verification, Model Checking, and Abstract Interpretation (VMCAI’13), LNCS, Vol. 7737, pp. 536–554, Springer, 2013.

**URL** [http://dx.doi.org/10.1007/978-3-642-35873-9\\_31](http://dx.doi.org/10.1007/978-3-642-35873-9_31)

Real-Time Embedded Software is an important class of safety-critical software systems. It plays a crucial role in controlling systems ranging from airplanes and cars, to infusion pumps and microwaves. Verifying the correct operation of RTES is an important open problem.

In this presentation, I will describe the START (Static Analysis of Real-Time Embedded Systems) project that I am leading together with Sagar Chaki at the Software Engineering Institute at Carnegie Mellon University. The focus of START is verification of safety properties (e.g., race conditions, mutual exclusion, and deadlocks) of periodic programs scheduled with Rate Monotonic Scheduling (RMS) policy. Such programs are common in automotive and avionics domains. I will describe our experience in building a Bounded Model Checker, called REK, for programs written for OSEK/VDX operating system, and our experience in using REK to verify properties of a robotics controller.

### 3.10 Certification for Medical Devices and Systems: An Overview and Challenges

*John Hatcliff (Kansas State University – Manhattan KS, US)*

**License** © Creative Commons BY 3.0 Unported license  
© John Hatcliff

**Joint work of** Hatcliff, John; Knight, John; Weber, Jens; Heimdahl, Mats

Medical devices and systems have long been an example of a safety-critical domain with many challenges related to risk assessment, regulatory policy, and certification. However, an aging population, innovations in mobile computing devices, increased reliance on integrated systems, and increased importance of storing and leveraging patient data are introducing a variety of strains and pressures on existing certification approaches.

In this talk, we give a summary of important certification-related issues in the medical device domain including the types of products certified in the domain, the most common regulatory processes and agencies, development and verification tools used in the medical device domain, and relevant standards for medical device certification. We conclude with a discussion of important trends and technologies within the medical device space that are giving rise to challenges that need the attention of researchers working in the areas of certification and verification.

### 3.11 Requirements Specification and Supporting Artifacts for an Open Source Patient-Controlled Analgesic Pump

*John Hatcliff (Kansas State University – Manhattan KS, US)*

**License** © Creative Commons BY 3.0 Unported license  
© John Hatcliff

**Joint work of** Hatcliff, John; Larson, Brian

**URL** <http://info.santoslab.org/research/pca>

The dynamic nature of the medical domain is driving a need for continuous innovation and improvement in techniques for developing and assuring medical devices. Unfortunately, research in academia and communication between academics, industrial engineers, and regulatory authorities is hampered by the lack of realistic non-proprietary development artifacts for medical devices.

In this talk, we give an overview of a detailed requirements document for a Patient-Controlled Analgesic (PCA) pump developed under the US NSF's Food and Drug Administration (FDA) Scholar-in-Residence (SIR) program. This 60+ page document follows the methodology outlined in the US Federal Aviation Administrations (FAA) Requirements Engineering Management Handbook (REMH) and includes a domain overview, use cases, statements of safety & security requirements, and formal top-level system architectural description. Based on previous experience with release of a requirements document for a cardiac pacemaker that spawned a number of research and pedagogical activities, we believe that the described PCA requirements document can be an important research enabler within the formal methods and software engineering communities.

### 3.12 Concerning the implicit DO-178C assurance case

*Michael Holloway (NASA Langley ASDC – Hampton, US)*

**License** © Creative Commons BY 3.0 Unported license  
© Michael Holloway

**Main reference** C.M. Holloway, “Towards Understanding the DO-178C / ED-12C Assurance Case,” in Proc. of the IET 7th Int’l Conf. on System Safety, October 2012, Edinburgh, Scotland.

**URL** <http://hdl.handle.net/2060/20120016708>

This informal discussion without visual aids describes ongoing work towards identifying and expressing explicitly the arguments contained in, or implied by, DO-178C, which implicitly justify the assumption that the document meets its stated purpose of “providing guidelines for the production of software for airborne systems and equipment that performs its intended function with a level of confidence in safety that complies with airworthiness requirements.”

### 3.13 Software verification in the medical domain

*Jozef Hooman (Radboud University Nijmegen, NL)*

**License** © Creative Commons BY 3.0 Unported license  
© Jozef Hooman

**Joint work of** Hooman, Jozef; Mooij, Arjan; Keshishzadeh, Sarmen; Albers, Rob

We present an overview of research activities to improve the verification and validation of medical systems at Philips Healthcare. In particular, we focus on the interventional X-ray systems of Philips. To reduce the test and integration phase of these systems and obtain a more efficient development process, the aim is to detect faults earlier by applying various modeling and analysis techniques. This includes executable models, domain specific languages, and formal methods.

### 3.14 Bridging the modeling/verification gap

*Jerôme Hugues (ISAE – Toulouse, FR)*

**License** © Creative Commons BY 3.0 Unported license  
© Jérôme Hugues

Model Driven Engineering (MDE) provides an appealing framework for supporting engineering activities, from early design phases to acceptance tests; going through refinement, architectural and functional design down to code generation and V&V efforts. Yet, certification activities may interfere with such process: traceability must be demonstrated, specific verification or validation activities must be performed, some of which are project or domain specific.

In this talk, I present current discussions on the part of the AADL standardization committee to enrich Architecture Description Language with a Constraint language. The objective is to increase the coupling between modeling and verification. By making the verification part of extended semantics rules of an ADL, we control the patterns used to describe the system, ensuring designers respect process requirements, but also integrate V&V as part of the modeling effort. Thus, it provides a lean approach to certification through MDE.

### 3.15 Opening up the Verification and Validation of Safety-Critical Software

*Hardi Hungar (German Aerospace Center – Braunschweig, DE)*

**License** © Creative Commons BY 3.0 Unported license  
© Hardi Hungar

**Joint work of** Hungar, Hardi; Behrens, Marc

**Main reference** In: M. Huhn, S. Gerken and C. Rudolph (eds.), Proc. ZeMoSS 2013, to appear

Smooth cross-border rail traffic is of important interest to commercial realizations of ETCS (European Train Control System). Starting from the hypothesis that the traditional way of developing software for safety-critical systems might be an obstacle to standardizing rail traffic, the ITEA 2 project openETCS has set out to pursue the idea of transferring an open-source development style to this domain, taking the EVC (European Vital Computer, core of the on-board unit) as a target.

The goal is to formalize the requirements in a functional model, derive, via design models, an implementation, and demonstrate how the verification and validation activities necessary for certifying a resulting product could be performed. All of this is to be done as an open-source project, employing only open-source tools. One of the main motives behind the approach is to use the potential of an open community to detect design and implementation flaws much earlier than the resource-limited inspection in a traditional development setting.

This talk discusses the challenges this new approach faces from the legal requirement of adhering to the standards, mainly the EN 50128 in this case, particularly with respect to verification and validation. This comprises the interpretation and application of the standard throughout all lifecycle phases for a open-source model-based development and qualification issues for personnel and tools.

### 3.16 Using Code Analysis Tools for Software Safety Certification

*Daniel Kaestner (AbsInt – Saarbrücken, DE)*

**License** © Creative Commons BY 3.0 Unported license  
© Daniel Kaestner

**Joint work of** Kaestner, Daniel; Ferdinand, Christian

**Main reference** D. Kästner, C. Ferdinand, “Efficient Verification of Non-Functional Safety Properties by Abstract Interpretation: Timing, Stack Consumption, and Absence of Runtime Errors,” in Proc. of the 29th Int’l System Safety Conference (ISSC’11), ISBN 9781618399922. Las Vegas, 2011.

In automotive, railway, avionics and healthcare industries more and more functionality is implemented by embedded software. A failure of safety-critical software may cause high costs or even endanger human beings. Also for applications which are not highly safety-critical, a software failure may necessitate expensive updates. Safety-critical software has to be certified according to the pertinent safety standard to get approved for release.

Contemporary safety standards – including DO-178B, DO-178C, IEC-61508, ISO-26262, and EN-50128 – require the identification of potential functional and non-functional hazards and to demonstrate that the software does not violate the relevant safety goals. To ensure functional program properties, automatic or model-based testing and formal techniques like model checking are becoming more widely used.

For non-functional properties identifying a safe end-of-test criterion is a hard problem since failures usually occur in corner cases and full test coverage cannot be achieved. For some non-functional program properties this problem is solved by abstract interpretation-based



static analysis techniques which provide full control and data coverage and yield provably correct results. Like model checking and theorem proving, abstract interpretation belongs to the formal software verification methods.

This talk focuses on static analyses of worst-case execution time, stack consumption, and runtime errors, which are increasingly adopted by industry in the validation and certification process for safety-critical software. First we will give an overview of the most important safety standards with a focus on the requirements for non-functional software properties. We then explain the methodology of abstract interpretation based analysis tools and discuss the role of formal verification methods in current safety standards. Using tools for certification requires an appropriate tool qualification. We will address each of these topics, report on industrial experience, and address open issues.

### 3.17 Towards an Effective Safety Demonstration Framework


*Peter Karpati (Institute for Energy Technology – Halden, NO)*

License  Creative Commons BY 3.0 Unported license  
© Peter Karpati

This talk will introduce our project which aims at assembling evolutionarily adaptable guidelines for an effective safety demonstration framework based on exploring state of the art and state of practice in the field.

### 3.18 Software Certification: Where is Confidence Won and Lost?

*Tim Kelly (University of York, GB)*

License  Creative Commons BY 3.0 Unported license  
© Tim Kelly

Given that we cannot prove the safety of software (in a system context) we are forced to wrestle with the issue of confidence in software certification. Some draw confidence from compliance with software assurance standards and believe this is sufficient, yet we don't have consensus in these standards. Some establish confidence through the process of constructing and presenting a software assurance case, but ignore the experience and "body of knowledge" provided by standards. Some (sensibly) use a combination of these approaches. Using our framework of 4+1 principles of software safety, this talk will discuss where and how in current safety-critical software development and assessment approaches confidence is typically won and lost. Based on this assessment, we describe how the activity and structure of an assurance case should best be targeted to explicitly address issues of confidence.

### 3.19 User Assembled Medical System of Systems

*Andrew King (University of Pennsylvania, US)*

**License** © Creative Commons BY 3.0 Unported license  
© Andrew King

**Joint work of** King, Andrew; Lee, Insup; John, Hatcliff

**Main reference** J. Hatcliff, A. King, I. Lee, A. Macdonald, A. Fernando, M. Robkin, E. Vasserman, S. Weininger, J.M. Goldman, “Rationale and Architecture Principles for Medical Application Platforms,” in Proc. of the 2012 IEEE/ACM Third Int’l Conf. on Cyber-Physical Systems (ICCPS’12), pp. 3–12, IEEE/ACM, 2012.

**URL** <http://dx.doi.org/10.1109/ICCPS.2012.9>

In safety critical domains, there is a typically a prime contractor that is responsible for integration and system-level verification and validation. In user assembled systems (such as plug and play medical systems), system modules are sold to users directly. These non-technical users then assemble these modules into (possible safety critical) systems of systems towards some purpose. We describe plug and play medical systems and associated certification challenges.

### 3.20 Three Challenges

*John C. Knight (University of Virginia, US)*

**License** © Creative Commons BY 3.0 Unported license  
© John C. Knight

In this talk, I will briefly summarize: (a) the concerns I have about the dependability of medical systems and why I think medical system dependability is more difficult than other domains, and (b) the concerns I have about standards together with some ideas about a “standard for standards.”

### 3.21 Certification of Medical Device Composition


*Brian Larson (Multitude Corp., US)*

**License** © Creative Commons BY 3.0 Unported license  
© Brian Larson

The Software Certification Consortium seeks to host “mock” certification evaluations of safety-critical systems dependent upon software. My presentation considers what artifacts should be part of submissions for mock certification, particularly evidence from analysis of architecture models of errors and behavior. Only formal proofs of safety and effectiveness of apps and medical devices they control will likely obtain regulatory approval.

### 3.22 Bayesian Probabilistic Approaches to Confidence are Impossible: The Need for a Baconian Approach (pace Jonathan Cohen)

*Tom S. Maibaum (McMaster University – Hamilton, CA)*

License  Creative Commons BY 3.0 Unported license  
© Tom S. Maibaum

Many have recognized the need for some notion of confidence in relation to safety and assurance cases. After all, a regulator has to have enough confidence in the case to prove a certification. After all, other domains also use such notions of confidence. These include the legal domain, where (at least in English law) judgements have to be made “on the balance of probabilities” (civil cases) and “beyond reasonable doubt” (criminal cases) are standards of confidence required of judges and juries about the guilt of the accused. Similarly, there is an implicit notion of confidence amongst scientists about the theories they use in their subject. Corroborative experiments raise this level of confidence, whilst negative result may lower the level of confidence (but not necessarily to 0, pace Popper). A number of philosophers/logicians/scientists have attempted to characterize this notion of confidence, including Bacon, and more recently, Carnap, Keynes, etc. More recently, Jonathan Cohen, in *The Probable and the Provable*, has demonstrated that the concept of probability underlying the concept of confidence, which he claims is Baconian, simply cannot be reduced to conventional Pascalian, frequency of occurrence, notions of probability. The argumentation basis for safety/assurance cases uses confidence as a tenet for the approach. Toulmin’s argument schemes present a form of inductive/scientific reasoning with explicit reference to confidence to justify the applications of a scheme. Interestingly enough, a scheme reduces to a deductive rule of inference when there is no uncertainty about its application.

### 3.23 Software Certification: The Return on Investment?

*John McDermid (University of York, GB)*

License  Creative Commons BY 3.0 Unported license  
© John McDermid

Certification costs money; it also has benefits, perhaps most importantly reduction in risk to system users and third parties, to which we give a value:

- The costs will be in manpower and other resources for testing, code inspections, formal verification, etc.;
- The benefit will be in terms of security breaches avoided, hazardous failures avoided, etc.;
- The value will be in terms of the assets protected (e.g. €Ms), or lives saved (perhaps monetised by multiplying by the VPF (value of preventing a fatality)), etc.

At its simplest, for there to be a positive return on investment (RoI) the value has to outweigh the cost. In practice, there are difficulties in determining RoI, for example mapping benefit to value due to uncertainties about how the software will be used. These difficulties are exacerbated prior to undertaking a software certification activity, e.g.:


- How do we predict costs?
  - How much will it cost to apply the technique to the software?
  - How much will it cost to rectify any flaws identified?
- How do we predict benefit before carrying out the activity, e.g. by simulation?
  - What flaws will we find?

- What flaws will we find that we would not find by other means?
- How do we predict value, as the benefits are really contingent?
  - Which assets will now be protected, should the attack we know could have been successful, but won't be now, actually occur? How likely is the attack?
  - How many lives will be saved, should the operational scenario we know could have been hazardous, but won't be now, actually arise? How likely is that scenario?

Whilst expressed in theoretical, or economic, terms there is a real practical issue here; when running a project, how much effort should be put into certification, and how do we know when to stop? (It is always possible to do more work and spend more money.) The talk will identify some of the inherent uncertainties in managing software certification and give some sanitised metrics on certification costs (mainly from the safety domain) which illustrate how much RoI can vary. It will also suggest some criteria by which we might judge any scheme to manage software certification so as to deliver positive RoI, and use this to stimulate a debate on how we evaluate the benefit of individual methods used in support of software certification.

### 3.24 Refinement may help for Certification

*Dominique Mery (LORIA – Nancy, FR)*

License  Creative Commons BY 3.0 Unported license  
© Dominique Mery

Joint work of Mery, Dominique; Singh, Neeraj Kumar

Formal methods have emerged as an alternative approach to ensuring the quality and correctness of the high confidence critical systems, overcoming limitations of the traditional validation techniques such as simulation and testing. Certification aims at assessing or demonstrating that a system complies with a collection of rules, regulations and standards defining the minimum requirements a system must have to be deployed, operated and dismissed in a safe way. It appears that formal methods may help in the process of certification... but with toil and with tools. We describe a methodology for developing critical systems from requirement analysis to automatic code generation with standard safety assessment approach. This methodology combines the refinement approach with various tools including verification tool, model checker tool, real-time animator and finally, produces the source code into languages using automatic code generation tools. This approach is intended to contribute to further the use of formal techniques for developing critical systems with high integrity and to verify complex properties, which help to discover potential problems. Assessment of the proposed methodology is given through developing a standard case study: the cardiac pacemaker. The pacemaker is a proposed case study of the grand challenge and we analyse the role of the refinement in the methodology, which is to be improved and experimented on other case studies. Finally, the refinement relationship provides a way to play with abstractions of software system under design. The general methodology promotes the use of refinement and the incremental development of models so called abstractions. As in classical engineering approaches, abstractions play a central role and we have provided additional tools for easing the communication between domain experts and method experts.

## References

- 1 D. Méry, N. K. Singh. Real-Time Animation for Formal Specification. In: Complex Systems Design & Management 2010, M. Aiguier, F. Bretaudeau, D. Krob (éd.), Springer, pp. 49–60. Paris, France, octobre 2010.
- 2 D. Méry, N. K. Singh. Critical systems development methodology using formal techniques. In: 3rd International Symposium on Information and Communication Technology – SoICT 2012, ACM, pp. 3–12. Ha Long, Viet Nam, août 2012.
- 3 mery:inria-00638473 D. Méry, N. K. Singh. Formalisation of the Heart based on Conduction of Electrical Impulses and Cellular-Automata. In: International Symposium on Foundations of Health Information Engineering and Systems (FHIES, 2011), Z. Liu, A. Wassylng (éd.), UMIST Macau. Johannesburg, South Africa, août 2011. conference 2011.
- 4 D. Méry, N. K. Singh. Technical Report on Formal Development of Two-Electrode Cardiac Pacing System. Rapport, février 2010.
- 5 D. Méry, N. K. Singh. Technical Report on Formalisation of the Heart using Analysis of Conduction Time and Velocity of the Electrocardiography and Cellular-Automata. Rapport, août 2011.
- 6 D. Méry, N. K. Singh. Technical Report on Interpretation of the Electrocardiogram (ECG) Signal using Formal Methods. Rapport, 2011.
- 7 D. Méry, N. Singh. EB2ALL- The Event-B to C, C++, Java and C# Code Generator. 2011, <http://eb2all.loria.fr>.

## 3.25 Certification Challenges for Software With Uncertainty

*Richard F. Paige (University of York, GB)*

**License** © Creative Commons BY 3.0 Unported license  
© Richard F. Paige

**Joint work of** Paige, Richard F.; Burton, Frank; Rose, Louis; Kolovos, Dimitrios; Poulding, Simon; Smith, Simon  
**Main reference** F.R. Burton, R.F. Paige, L.M. Rose, D.S. Kolovos, S. Poulding, S. Smith, “Solving Acquisition Problems Using Model-Driven Engineering,” in Proc. of the 8th European Conf. on Modelling Foundations and Applications (ECMFA’12), LNCS, Vol. 7349, pp. 428–443, Springer, 2012.  
**URL** [http://dx.doi.org/10.1007/978-3-642-31491-9\\_32](http://dx.doi.org/10.1007/978-3-642-31491-9_32)

This talk presented several different types of software applications – all considered safety-critical or safety-related – that exhibited uncertainty in some sense. By uncertainty, we meant “openness” or “unpredictability of behaviour”, wherein some behaviours were emergent at run-time. One application, based on [?], made use of evolutionary algorithms to calculate optimal solutions to decision making problems in the enterprise (e.g., acquisition of new equipment, new training of personnel, development of new policies). The second application exhibited two types of uncertainty: in terms of data sources (which could change at run-time, e.g., switching from a batch or cached data source where data quality was guaranteed to a real-time error-high data source) and in terms of user-programmability (where users could encode new behaviours in the system by developing new execution patterns). The two applications were illustrated and a set of certification challenges were identified. These challenges included gathering of evidence, auditing evidence, assessing the quality of evidence, and process issues (e.g., ensuring that sufficient assurance was obtained for specific types of changes).

### 3.26 Models and Certification

*Andras Pataricza (Budapest Univ. of Technology & Economics, HU)*

License  Creative Commons BY 3.0 Unported license  
© Andras Pataricza

Model Driven Engineering (MDE) becomes the main trend in critical embedded systems design. The presentation focuses on the fundamental question: How the different models and artifacts can be reused for supporting the certification process? A novel solution for traceability support is presented. The use of ontologies for the formalization of text documents like standards is proposed. Dependability analysis is addressed in the context of MDE. Finally, an approach for the empirical validation of models and underlying assumptions is proposed.

### 3.27 From Tool Qualification to Tool Chain Design

*Jan Philipps (Validas AG – München, DE)*

License  Creative Commons BY 3.0 Unported license  
© Jan Philipps

As engineering disciplines become more mature, more emphasis is put not only on the way of working, but also on the tools used. In safety standards, we can observe a similar development. Earlier standards put only little emphasis on tool use, perhaps roughly demanding a separate argument that each tool individually be “fit for use.” Recent standards, such as the ISO 26262, take a more holistic viewpoint. Not only tools themselves, but also their use in the development process of the project must be analyzed, risks identified and only if necessary, further tool qualification measures employed.

Qualification measures typically are a combination of extensive tool testing and an analysis of the development process used for the tools. These approaches tend to be rather costly. Some tool vendors give support in the form of so-called qualification kits, which, for instance, include ready-to-run test cases. In practice, however, development tool chains consist not only of commercial tools, but also of open source or bespoke tools. The cost of qualifying all these tools would be prohibitive.

However, the holistic viewpoint taken in the ISO 26262 opens up new possibilities in building trust for tools and tool chains. Instead of just looking at isolated tools, it is possible to reduce qualification demand through tool diversity or through error avoidance and detection mechanisms within the tool chain. This is not unlike systems design, where safety is achieved by a combination of architectural choices and component reliability.

The talk presents this shift from tool qualification to tool chain design and gives some examples from an industrial project.

### 3.28 Cloud Security: Information Segregation and Data Privacy

*Julia Rubin (IBM – Haifa, IL)*

License © Creative Commons BY 3.0 Unported license  
© Julia Rubin

As cloud computing gains popularity both in private and enterprise sectors, it is only reasonable that customers expect guarantees for secure care of their data. In this talk, we focus on certification for cloud security, specifically information segregation and data privacy. We motivate the need for such certification and discuss its main differential factors from the more established certification procedures, e.g., those in the domains of safety-critical systems and, recently targeted, sensitive data protection in the healthcare and financial industries.

One of the major challenges for security certification is to verify not only that the system does what it is expected to do, but also that it does not do what it is not expected to. To address this challenge, we propose a specification-based runtime verification approach which relies on model checking, model-based testing, monitoring and run-time data analysis techniques. We believe that this approach is significantly more powerful than conventional testing and more practical than traditional formal verification for verifying security properties of a system.

### 3.29 Logic and Epistemology in Assurance Cases

*John Rushby (SRI – Menlo Park CA, US)*

License © Creative Commons BY 3.0 Unported license  
© John Rushby

Any assurance case comes down to two kinds of questions: how complete and accurate is my *knowledge* about aspects of the system (e.g., its requirements, environment, implementation, hazards) and how accurate is my *reasoning* about the design of the system, given my knowledge.


The first of these is a form of *epistemology* and requires human experience and insight, but the second can, in principle, be reduced to *logic* and then checked and automated using the methods of formal verification. (There are “inner” epistemic questions here, concerning correctness of the verifier, but we’ll postpone those for the time being.)

The distinction between concerns that are epistemic or logical in origin is exactly that underlying the traditional partitioning of assurance into Validation and Verification (V&V). To some extent, it is possible to trade the two kinds of concerns (e.g., strong fault models allow simple fault-tolerant implementations: this reduces logic doubt about correctness of the implementation but increases epistemic doubt about verity of the model).

We propose that reducing epistemic doubt should be the main focus in assurance cases, and discuss ways in which this might be achieved.

### 3.30 Model-Based Development and Functional Safety

*Bernhard Schaetz (fortiss GmbH – München, DE)*

License  Creative Commons BY 3.0 Unported license  
© Bernhard Schaetz

Model-based development has demonstrated its benefits in the automotive industry in improving development time and costs. Being a de-facto standard in automotive software development, it has found explicit acknowledgement as relevant technique by ISO 26262.

Its main advantages lie in front-loading of quality assurance techniques and automation of implementation steps. We show how this can also contribute to functional safety by explicating assumptions about platform and environment, by enabling high degree of precision as well as a scalable degree of detail, by supporting in-depth understanding of assumption, by supporting correctness of design and implementation, and by enabling automation of analysis and synthesis.

### 3.31 Software Certification Challenges in the Nuclear Power Domain

*Alan Wassying (McMaster University – Hamilton, CA)*

License  Creative Commons BY 3.0 Unported license  
© Alan Wassying

The current state of (software) certification in the nuclear power industry, its major challenges and a brief comparison with other domains in which systems are safety critical. The primary examples are drawn from a Canadian perspective.

### 3.32 Certification of Medical Information Systems – A paradigm shift: from devices to systems, from functions to data

*Jens H. Weber (University of Victoria, CA)*

License  Creative Commons BY 3.0 Unported license  
© Jens H. Weber

Medical information systems (MIS) play a pivotal role in modern health care systems. They perform increasingly critical functions with respect to human safety, privacy and security. A significant number of adverse events has been associated with failures in MIS and this has resulted in increasing calls for their certification and regulation. Unfortunately, there is much confusion and little agreement on how to certify these systems. Existing paradigms as for example used for certifying traditional forms of medical devices do not readily apply. My presentation contrasts unique aspects of MIS from those found in other classes of critical computer-based systems. I point out why current certification techniques are insufficient with respect to achieving the overarching certification objectives. I then describe a paradigm shift that is needed in order to arrive a more effective certification program for MIS. Finally, I indicate research challenges and opportunities in this area.



### 3.33 Software certification in aeronautics

*Virginie Wiels (ONERA – Toulouse, FR)*

License © Creative Commons BY 3.0 Unported license  
© Virginie Wiels

This talk presents an overview of avionics software certification, including the process, the certification authorities, the different criticality levels. It describes the main principles of DO-178, which is the certification standard in this domain. It briefly mentions the recent update of DO-178 and in particular the Formal Methods Technical Supplement. It also lists future challenges for software certification.

#### References

- 1 Guidance for Using Formal Methods in a Certification Context. Duncan Brown, Hervé Delseny, Kelly Hayhurst, Virginie Wiels. ERTSS 2010 May 2010, Toulouse, France

### 3.34 Some experience and remarks on security certification at industry

*David von Oheimb (Siemens AG – München, DE)*

License © Creative Commons BY 3.0 Unported license  
© David von Oheimb

In my overview talk I share my experience with IT security certification at Siemens Corporate Technology. This type of activity is relatively rare in industry, for a number of reasons backed up by several examples. I briefly introduce the Common Criteria (CC), list several types of involvement of our group in their use, at examples like digital tachographs, an airplane SW distribution system, smart card processors, and smart metering gateways.

## 4 Overview of Working Groups

### 4.1 Challenges: Compositional Certification

In virtually all domains of modern computing systems, software size and system complexity are growing rapidly. Increased scale and complexity are straining current methods for system development, and in particular, methods for ensuring safety and for certification. A general engineering principle for managing complexity is to (a) decompose a system down into multiple smaller components that can be worked with individually through multiple phases of development, and (b) integrate components in later stages of development to form a complete system. Decomposing systems into components can also lead to cost reductions and decreased development time when components are *reused* across multiple systems. Unfortunately, the effectiveness of these strategies is limited in the context of certified systems, because almost all certification regimes for safety-critical domains certify complete systems – not system components. This state of affairs does not result from a lack of insight or interest on the part of industry or certification authorities. Rather, it is driven by the fact safety issues often arise due to incorrect context assumptions or unanticipated interactions between components, between software and associated hardware, and between a functioning system and its environment. Therefore, reasoning about safety, using current practices, is most easily carried when one has as much context information as possible. Of course, context

information is maximized when working with *all* system components in an integrated state, i.e., when working with the system as it will actually be deployed.

A seminar working group considered the issues related to component-wise development in the context of certified systems. In particular, the group was interested in the issue of *compositional certification*: a process by which individual components could be certified so that when components were assembled into complete systems, certification activities and arguments would not require a full assessment of all components implementations but instead could rely, to a large extent, the certification outcomes of the individual components. Such a process would allow both component implementations *and their certification artifacts/results* to be reused across different system implementations.

#### 4.1.1 Current practices and trends motivating the need for compositional approaches to certification

In addition to the general challenge of dealing with systems of increasing size and complexity, the working group noted several important trends in system development that are motivating the need for compositional approaches to certification.

The notion of an integrated “system of systems” (SoS) is a category of systems that are increasingly prevalent and particularly challenging to certify. While definitions vary, a SoS is generally understood to be a collection of systems that can each function as a stand-alone system in some capacity but are integrated typically by some network-centric architecture to achieve new mission capabilities not obtainable from the individual systems. Small to mid-scale examples include modern automotive and avionics systems, where many microcontrollers, sensors, and actuators interconnect via a communications infrastructure that allows information from each set of sensors to calculate actions of actuators across the entire system. In larger examples, military command and industrial control systems are increasingly moving from stand-alone, monolithic designs to integrated platforms.

The concept of *software product line engineering* has proven to be very effective in a number of large organizations. Product line engineering is applied when one has a family of similar systems. An effort is made to (a) identify functionality that is common across multiple systems within the family, (b) design and implement components that provide that functionality, and (c) systematically design systems so that common components can be reused across multiple systems within the family. The end result is that costs are saved and development time is decreased by reusing components across multiple systems. While the product line concept is most often applied within a single organization, component reuse across organizations is facilitated when component interfaces are clearly defined or standardized and when a commodity market develops for component implementations.

Architecting *computational platforms* is becoming a common approach for achieving reuse. In this approach, a run-time environment, common services, and frequently used application components are shared between many stakeholders. To develop an application, one need not develop system functionality from the ground up; instead, one focuses on developing application logic using the shared services and application building blocks to produce an application that executes in the provided run-time environment. Smart phone platforms such as the iPhone and Android are one of the most prominent examples in the consumer space. A platform approach can also encourage innovation since, by removing the need to build many parts of the system, it allows more people with less capital to enter the market and contribute ideas.

While integrated systems in general are not common in the medical device space as they are in e.g., avionics, automotive, etc., several talks in this seminar have presented research

results related to the concept of a *medical application platform* (MAP) [4]. A MAP is a safety- and security- critical real-time computing platform for (a) integrating heterogeneous devices, medical IT systems, and information displays via a communication infrastructure and (b) hosting application programs (i.e., *apps*) that provide medical utility via the ability to both acquire information from and update/control integrated devices, IT systems, and displays.

In summary, the compositional and/or component-wise development present in all of these types of systems mentioned above cannot currently be adequately aligned with existing certification regimes due to their lack of support for compositional certification arguments.

#### 4.1.2 What would we like to be able to do that we can't do now?

The group identified several specific capabilities, technologies, and products that it believed would be important for moving toward more compositional approaches to building safety- and security-critical systems.

- (pre)-Certifiable high-assurance platforms that organizations can purchase and re-use.
- Better technologies (or more widely applied technologies) for ensuring partitioning in terms of space and time. This helps ensure that composition will not negatively impact safety.
- Approaches for making compositional safety arguments
- Formal capture of functional and non-functional properties to enable mechanical reasoning (e.g., checking interface compatibility, checking interface compliance) either a priori or at run-time/composition time.

#### 4.1.3 What are the barriers (technological, standards, regulatory policy, political, social) that are hindering an advance toward a solution?

Regarding technology barriers, there is a need for better methods for dealing with *emergent behaviors* – behaviors that are not present or at least cannot be understood well in individual components but only arise when components are integrated. Specifically, better techniques are needed for:

- a priori recognition (or the possibility of) hazardous emergent behaviors – in essence, there is a need to develop compositional approaches for hazard analysis and risk assessment,
- engineering principles for minimizing unanticipated interactions and emergent behaviors,
- better and more systematic post-deployment means of detecting harmful emergent behaviors (you don't know what you don't know, so at least try to check after the system is deployed if something is going wrong).

Proposition compositional development relies on well-defined and precisely specified interfaces. There is need for better methods of (a) capturing interface properties that specify *non-functional* “contracts” including quality of service (QoS) and real-time properties, (b) capturing effects that a component can have with the environment (to detect possible interactions through the environment). As an example of the later property, magnetic resonance imaging (MRI) machines may impact the environment by causing interference with wireless communication of other medical devices in close proximity. Better technology is also needed for formally capturing modal/state behavior and security policies of components. In the vision for medical application platforms introduced above, system composition/integration in a health-care delivery organization – after system platforms and associated apps and devices are certified and deployed. Therefore, there is a need for technology that would allow the

run-time system of the medical platform itself to dynamically moderate composition so as to only enable those compositions that will produce a safe system.

Regarding business barriers, it was noted that some companies prefer proprietary systems so that they can lock small manufactures out of the market. Thus, while society as a whole would benefit from more open, component-based approaches, many companies are not interested in defining the interfacing standards necessary for achieving this vision. Moreover, systems composed from heterogeneous components (components from different manufacturers) often lead to questions about liability, i.e., which manufacturer is liable when a system fails.

Regarding regulatory barriers, allowance for compositional approaches in existing certification and regulatory guidelines is minimal. In the medical space for example, there are currently no guidelines for how systems following the notion of “medical application platform” should be regulated. To some degree this is justified because the community has not yet arrived at a convincing approach for demonstrating safety in the compositional setting. In the medical space, interoperability and compositional approaches are also hampered by the lack of widely implemented interoperability standards. In avionics, Integrated Modular Avionics (IMA) [2] provides some notion of reuse, but only supports static configuration; that is, changing configurations requires new certification. Seminar attendees shared the perspective that some in the broader community believe that IMA will reduce costs, but others say that the use of IMA increases the difficulty of making safety arguments – to the extent that the cost reductions achieved are not as significant as they are “advertised” to be.

#### 4.1.4 What evidence is there that success is possible?

Participants shared several notions of reuse or compositionality in certified systems.

- In the security domain, one of the original motivations for the Multiple Independent Levels of Security (MILS) architecture [1] was to promote a commodity market of reusable components for security systems to be certified according to various protection profiles within the Common Criteria [3].
- In the avionics domain, guidance for Integrated Modular Avionics (DO-297) describes how a particular architecture supports flexibility, modularity, reusability and interoperability.
- ISO 26262, which addresses functional safety for road vehicles, includes the notion of “safety element out of context” (SEooC) which allows the statement of assumptions and guarantees for a particular element (informally, a sub-system) whose development is being carried out by, e.g., a sub-contractor.
- The FAA Advisory Circular on Reusable Software Components (AC 20-148) provides a means for reusable software component (RSC) developers, integrators, and applicants to gain: FAA “acceptance” of a software component that may be only a part of an airborne system’s software applications and intended functions, and credit for the reuse of a software component in follow-on systems and certification projects, including “full credit” or “partial credit” for compliance to the objectives of RTCA/DO-178B, Software Considerations in Airborne Systems and Equipment Certification.
- UK Defense Standard 23-09, standardizes interfaces within the Generic Vehicle Architecture, an open source architecture specification with well-defined interfaces that aims to encourage reuse in military ground vehicles.
- The cross-domain functional safety standard IEC 61508, allows the notion of a “pre-certified” component that can be integrated to a 61508-certified system while reusing some of the certification effort. For example, a Green Hills produces a pre-certified IEC 61508 Safety Integrity Level 3 real-time operating system kernel – the Platform for Industrial Safety (PIS). IEC 61508 is an international standard for the functional safety

of electrical/electronic, programmable electronic systems (PES) and is well established in the industrial process control and automation industry. Because IEC 61508 serves as the meta-standard for a range of industries and published standards, the Platform for Industrial Safety is directly applicable to railway (CENELEC EN 50128), medical (IEC 60601), nuclear (IEC 61513), process control (IEC 61511), and automotive (ISO 26262). Such pre-certified components are documented in 61508 by a “Safety Manual”.

These existing approaches could be considered to be precursors to an eventual more robust and rigorous approach to compositional certification. Techniques such as self-verifying software, runtime verification, proof-carrying code, model composition focused on system verification, modular assurance cases [5] may be key enabling technologies.

## References

- 1 C. Boettcher, R. DeLong, J. Rushby, W. Sifre. The MILS Component Integration Approach to Secure Information Sharing. Presented at the 27th IEEE/AIAA Digital Avionics Systems Conference (DASC), St. Paul MN, October 2008.
- 2 P. Conmy, M. Nicholson, J. McDermid. Safety assurance contracts for integrated modular avionics. Proceedings of the 8th Australian workshop on Safety critical systems and software, Volume 33, pp. 69–78, 2003.
- 3 R. DeLong, J. Rushby. A common criteria authoring environment supporting composition. Proceedings of the 8th International Common Criteria Conference, 2007.
- 4 J. Hatcliff, A. King, I. Lee, A. Macdonald, A. Fernando, M. Robkin, E. Vasserman, S. Weinger, J. Goldman. Rationale and Architecture Principles for Medical Application Platforms, Proceedings of the 2012 International Conference on Cyber-Physical Systems, pp. 3–12, April, 2012.
- 5 T. Kelly, S. Bates. The Costs, Benefits, and Risks Associated With Pattern-Based and Modular Safety Case Development. Proceedings of the UK MoD Equipment Safety Assurance Symposium 2005, October 2005.

## 4.2 Challenges: Education and Challenge Problems

The seminar included significant discussions on education related to certification, and in particular, on the use of challenge problems and realistic case studies in education.

In prepared remarks given at the Information Technology and Innovation Foundation, April 12, 2012, Washington, DC, Thomas Kalil listed several properties/goals of Grand Challenges: (1) they can potentially have a major impact in the domain, (2) they should be ambitious but achievable, (3) they should be compelling and motivating, (4) they should be focused – must know when they have been achieved, and (5) they should drive innovation and advance technology. In addition, we note that challenge problems should be “research intensive”. That is, if you look at the definition of a grand challenge it should be obvious that the emphasis is on the research involved in the challenge.

The community associated with this Dagstuhl Seminar has created several challenge problems in the medical domain (e.g., the Pacemaker, and Patient-Controlled Analgesic Pump described in talk abstracts appearing earlier in this report). For example, Boston Scientific (through Brian Larson) released into the public domain a sanitized requirements document for a 10 year old pacemaker. Brian also worked with an Electrical and Computing Engineering Capstone class at University of Minnesota to design a hardware reference platform. Mark Lawford manufactured and sold 50 modded units. The specific challenge was to use the natural language requirements document as the basis for a formal approach to building a

pacemaker Currently more than ten different prominent institutions tackling the challenge in coursework. Over twenty research papers have been published that use pacemaker artifacts. A Dagstuhl Seminar on the Pacemaker Challenge has been approved for Feb 2–7, 2014.

Assessing the pacemaker challenge against Kalil’s criteria for research challenges, it has all the attributes of a Grand Challenge but one: it is not focused enough. There was a lack of specificity regarding the research objectives and expected artifacts to be produced. The lack of focus has led to the effort being less of a driver for true innovations in verification techniques. Teams tended to just take small bites or slices of the problem and address issues that existing tools were already able to handle well. We must make sure not to repeat that mistake in future challenges. Another issues is that the challenge does not run itself – there are real resourcing issues. Experts really need to be available to provide background domain knowledge, occasional support for the hardware, and evaluation of proposed solutions.

However, as a driver for innovative pedagogy, the Pacemaker Challenge has been unexpectedly successful. For example, the capstone undergraduate class at McMaster (run by Mark Lawford), the third year class on software development for ECE and Mechatronics Engineers (Alan Wassyng) at McMaster, courses at the University of Pennsylvania, etc. all use the Pacemaker materials. In addition, the SCORE student competition activity at the International Conference in Software Engineering in 2009, the Pacemaker artifacts were used by four 4 groups, one of which won the prize for best use of formal methods.

There are major components of these grand challenges that are reflective of state of the art and do not require the final research breakthrough that is being sought in the Grand Challenge. They typically involve hot topics in their specific domains. Tremendous effort is being expended in tackling these problems, and we can borrow from that effort to reinforce our education components in these domains. We think we can bridge the gap between state of the art/practice and general practice by disseminating these problems with guidance material.

The audiences who might benefit from an increased emphasis on *educational* material associated with challenges like Pacemaker are quite diverse: (a) students in safety/security-critical systems, (b) industry engineers who we want to orient to “new/improved” techniques that we are proposing regulatory / certification agencies, and certification community members working on research topics to advance the state of the art in certification / regulatory science.

If one focuses on the pedagogical/education space with challenges problems, there slightly different goals to consider. The problem should be complex enough to require sound techniques for their solution, when supporting projects the problem should be complex enough to need groups rather than individuals, to add realism it is ideal to have both hardware and software components, the challenge should be “solvable” in a span of time related to semester length or academic year length (e.g., 8 months for a year-long capstone project, and can be scaled down so that a subset that can be solved in four months), and finally, there needs to be appropriate domain background material, requirements, etc.

What are some specific things that should be offered:

- Requirements specification – should this be natural language, formal, or both?
- Dependable hardware platform at reasonable cost
- Hardware manual(s)
- Clear goals
- Guidance for development of solutions
- Mechanisms for evaluating solutions
- Domain specific background material
- Certification related background material

- Details on a slice through the system
- Wiki with FAQ and discussion groups
- Competitions

This type of challenge problem activity has several benefits to students. They get to tackle a project that is representative of “real” projects, they (hopefully) get excellent support material, they get guidance “best practices” and on how to “do it right” for a realistic problem, and they are typically excited to have it appear on their resumes. Pre-packaged educational material for challenge problems also has numerous benefits for instructors. The details of a “case study” is given to them with all supporting material including availability of a hardware platform. They get support through a Wiki. They get the benefit of cross-fertilization of a common problem tackled in many countries. When the supplied pedagogical material includes evaluation criteria and guides, they instructors do not need to work so hard on this themselves. Finally, as a challenge problem’s use increases, “frequently asked questions” and Wiki contents accumulate which provides more resources for future issues.

### 4.3 Challenges: Security

IT security is of increasing importance in various areas as electronic commerce and governance, any kind of access control systems, privacy of personal and in particular medical data, and information segregation in the cloud. IT-security is a cross-cutting concern in most systems. Moreover, with the tighter integration of formerly stand-alone systems to systems of systems and the enhancement of functionality, functional safety of systems relies on IT security. Incidents emphasizing the vulnerability of safety-critical systems by security attacks have been reported e.g. from the transportation domain and medical systems.

Whereas safety considerations on software have a tradition since decades as they were required with the first usage of programmed functionality in safety-critical systems, the need for specific and systematic IT-security analysis and design methods has been recognized later. Initially a lot of methods have been transferred from safety, in particular approaches to assure functional correctness apply for both areas. However, the requirement to specifically identify potential loopholes usable by malicious attackers is a distinctive feature of security analysis. Standards, in particular the Common Criteria, are widely accepted nowadays. Due to practitioners, the standards have proven beneficial, but need to be applied in a sensible way.

The working group stated the following *general* challenges of software certification for IT security: How can manufacturers actually prove that their products are secure, rather than just declaring it? How can the proliferation of profound new security techniques into industrial practice be accelerated? Due to short life cycles of many applications and missing compositionality, security is approved only for a short period of time. Thus approaches to continuous (re)certification are needed. Many standards can be improved in order to be more prescriptive, more up to date, and document best practices. Security and safety concerns shall be unified to support correct-by-design, safe-by-design, and secure-by-design.

As a particular issue, that shall be addressed in further research, *compositionality in security analysis and design* was identified: Obviously assembling secure components is not sufficient to produce a secure system since security – as safety – is an emergent system property. Layering in the architectural design and considering security a first class issue may help, but will not solve all issues. Further classification is needed in order to identify architectural patterns that support particular security properties.

Awareness and education is still an issue, because even software developers who graduated recently are not aware of security and do not know about methods to design and implement secure software. Even in large IT projects an (independent) security expert in the design team is often missing.

Last but not least the perception of IT security in the public audience will be a key factor not only for future research activities on IT security, but even more for the success or failure of several sectors within IT industries like social media or the cloud that rely on commonly accepted security policies.

#### 4.4 Challenges: Tool Qualification

Tool qualification is the process by which certification credit may be claimed for the use of a software tool. The purpose of tool qualification is to provide sufficient confidence in the tool functionality so that its output may be trusted. Tool qualification is, therefore, a significant aspect of any certification effort. The tool qualification working group attempted to identify initial needs and challenges related to use of software tools in certification efforts.

One way that we may obtain confidence in the output of a software tool is to provide some independent verification of the tool output. This may be accomplished by a manual review (if feasible) or by using an independent tool of equivalent functionality and comparing the outputs. Qualification is required whenever a software tool is used to eliminate, reduce, or automate a software life cycle process without the tool's output being verified.

Many different types of tools may be used in a software development process that could impact the correctness of the software. For example:

- Requirements engineering tools for eliciting, capturing, and specifying requirements
- Traceability tools for managing the design rationale and connections between software life cycle data
- Design tools for transforming requirements and constraints into design models
- Transformation tools such as code generators and compilers
- Test generation tools for producing test cases from requirements specifications
- Verification tools for analyzing design models or code to determine compliance with requirements
- Tools for generating and checking software configuration data

However, tools may be broadly categorized for qualification purposes (as in DO-178B) according to how they may impact software correctness:

- Tools that could introduce a fault into the software (development or transformation tools)
  - Tools that could fail to detect an error already present in the software (verification tools)
- Tools may be further categorized according to the criticality of the product software that they are generating or verifying.

In general, the qualification requirements for a development or transformation tool are much more stringent than for a verification tool. A reasonable approach in some cases is to qualify an independent verification tool to check the output of an unqualified development tool. For example, a complex tool may be needed to generate a schedule; however, tool for checking the correctness of a given schedule is relatively simple.

##### 4.4.1 Needs

The working group identified a number of needs in the area of tool qualification.



Software tools are used in development processes to automate life cycle activities that are complex and error-prone if performed by humans. The use of such tools should, in principle, be encouraged from a certification perspective to provide confidence in the correctness of the software product. Therefore, we should avoid unnecessary barriers to tool qualification which may inadvertently reduce the use of tools that would otherwise enhance software quality and confidence.

Most software tools are not used in isolation, but are used as part of a complex tool chain requiring significant integration effort. In general, these tools have been produced by different organizations. We need to develop better and more reliable methods for integrating tools from different vendors (including university tools, open source tools, and commercial tools).

A given software tool may be used in different application domains having very different requirements for both certification and tool qualification. Furthermore, the methods and standards for tool development varies across domains. Consistent qualification requirements across different domains would simplify the process.

#### 4.4.2 Barriers

The working group discussed several barriers that may inhibit tool qualification today.

Complexity is a barrier to tool qualification that may manifest itself in several ways. The input/output space of the tool may be complex, as is the case for compilers and code generators. For other tools, such as model checkers, the results produced are difficult to verify. Another source of complexity may be the algorithm used by tools, as in the case of transformation tools that rely on artificial intelligence techniques, evolutionary methods, or non-deterministic algorithms.

The platform on which a tool executes may present barriers to qualification. For example, the use of a virtual machine (VM) may introduce uncertainties in how a tool executes that may be hidden from view, or may vary when a different VM is used. Reliance on COTS libraries outside the scope of the software tool itself is another source of uncertainty if these libraries are not completely identified and specified.

Many valuable and effective software tools are developed using less than rigorous tool development processes. This is true for many tools developed in the university environment where the research agenda is a higher priority than qualification requirements. This can be an impediment to the subsequent qualification of the tool for use in a certification process.

#### 4.4.3 Roadmap

A number of steps were discussed that could be part of a roadmap for improving the tool qualification process.

Improved methods for tool analysis could be developed. For example, tool chain integration has been identified as a significant issue. Perhaps a HAZOP-type approach could be used to assess the potential errors introduced in integration. Another approach would be to identify common patterns of tool errors and ways to control or avoid these errors.

In the area of tool construction, new methods for providing evidence of correct tool operation could be developed. For example, a tool could provide evidence as part of its installation or at runtime or correct operation. Another approach is to focus efforts on tool architectures based on the idea of a complex (but unqualified) transformation that is checked by a simpler (qualified) verification tool.

Another new and promising approach to tool qualification is found in the use of formal methods to verify software tools. The CompCert compiler project (<http://compcert.inria.fr/>) is an example of this approach.

A final need is a thorough comparison of the qualification viewpoints and demands of the different domains. This should include an assessment of the point of view of different regulatory bodies.

Several useful references for qualification in the avionics and nuclear domains were identified:

- *DO-330/ED-215: Benefits of the New Tool Qualification Document*  
<http://www.adacore.com/knowledge/technical-papers/do-330-ed-215-benefits-of-the-new-tool-qualification-document/>
- *Licensing of safety critical software for nuclear reactors: Common position of seven European nuclear regulators and authorised technical support organisations*  
<http://www.hse.gov.uk/nuclear/software.pdf>

## 4.5 Intellectual Basis for Certification & Confidence

The current intellectual basis for certification of software intensive systems and how regulators gain confidence in the safety and reliability of systems is largely based upon the application of process oriented standards. Currently there is growing consensus that the current status quo has to change. A working group at the seminar considered the related questions (i) what should form the intellectual basis of certification? And (ii) How do we gain sufficient confidence in software intensive systems?

### 4.5.1 What is happening in today's world and into the future that is driving the need for change?

- There is a lack of repeatability of certification results. Currently the outcome of a certification is overly dependent upon the evaluator(s), or, in some cases similar cases before the same evaluator results in different outcomes.
- Regulators and developers do not know how to evaluate different types of evidence and understand how they may be combined to achieve a desired level of confidence in the system. Further they do not know what evidence to collect or do not agree about relevant values of particular types of evidence. There are strong disagreements even among recognized experts on these points.
- There is unnecessary variation across application domains in terms of both practice and regulation. The goals are largely the same in all sectors – achieving tolerable risk in a software intensive system – yet there is wide variation on what constitutes acceptable evidence in different domains. This makes it more difficult to share expertise in both development and evaluation as well as methods and tools.
- Often developers and certifiers feel that there is wasted effort that does not add value since they are unsure sure how things contribute to adding confidence. There is a desire to eliminate busy work and focus efforts on those aspects of development and evaluation that make a real contribution achieving tolerable risk. Currently it is unclear how to determine whether effort is wasted or not.
- There is a gross association between Dependability Assurance Levels (DALs) or Safety Integrity Levels (SILs) to moderate confidence. It is not clear that such coarse grained classification of systems and their respective appropriate evidence is the best method.

- The types of systems we now want to build are different than they were in the past. They are now continually evolving, adaptive systems that may be rapidly reconfigurable during operations.
- It is not always clear how much confidence is “enough”. There are sometimes technical issues in making this determination, but there are others such as the societal acceptance of the trade offs involved in the issues of risks vs. confidence vs. benefit vs. cost. It is possible to be overly cautiousness because of fear of the unknown or on the other hand be too quick to accept what may be a largely unknown risk.
- The difference between normal engineering (e.g. development of a minor revision of a well understood product) vs. radical design (development of a novel, first of a kind product) is not always understood. In the current practice of the engineering of software intensive systems, there is often no clear basis for distinguishing between them.
- The amount of safety-critical software is increasing substantially. Functions for which people were once responsible are being transferred to software systems. This may results in an inadequate ability to control the rate of change in requirements.
- In order to reduce costs there is trend towards the increased use of commercial software and hardware in critical systems. The incorporation of Software Of Unknown Providence (SOUP) into critical systems imposes further challenges for certification. This is but one example of how systems may be evolving in ways that are antithetical to safety-critical use.

To summarize, what would we like to be able to do that we cannot do now with the current basis of certification is to solve the problems posed above in a systematic way.

#### 4.5.2 The Barriers

What are the barriers (technological, standards, regulatory policy, political, social) that are hindering an advance toward a solution? Commercial interests dictate that in order to preserve intellectual property such as trade secrets, companies are reluctant to provide full source code and system documentation, preferring “black box” certification. Further entrenched cadres are invested in the current practice viewing their knowledge of how to navigate current opaque certification regime as a competitive advantage and a barrier to entry for start-up competitors. There is little historic data available on certification. Correlating the data that does exist with causal factors of success/failure is very difficult.

There is a widely held idea in academia that verification is equal to certification. This demonstrates a lack of understanding of industry practice by academics resulting in research into and teaching of methods and tools that do not scale to industry problems. The view of the working group was that currently there is inadequate education for a basis for certification. Even what we know works well is not known widely enough and taught even less. The result is that we are graduating engineers that are not recognizing that confidence is an issue that needs to be addressed. When it is address there was concern in the working group that we are often using the wrong mathematics for reasoning about confidence. A view was expressed that quantitative arguments about the reliability of software intensive systems is misleading since it is extremely difficult to get accurate failure rates for software.

#### 4.5.3 Evidence that success is possible

What elements of potential solutions exist in research, existing standards, existing technology? Some domains, such as aerospace, achieve good results although the reasons for the success are not necessarily known and the costs are often high. There is growing recognition of

the problems involved in the certification of software intensive systems and this is resulting in improvements in the capabilities of tools to support aspects of certification. Domains that previously have not had knowledge in the development and certification of software intensive systems are gaining it and there are some signs of movement from the past process-orientation certification regimes to product-orientation methods. There is a current trend towards company standards including explicit confidence arguments and a number of case studies are being performed. There is work being done of the development of reference designs to normalize engineering in areas such as infusion pumps and there are efforts to provide engineering cookbooks such as the FAA Requirements Engineering Management Handbook.

## 4.6 Methods for Developing Certifiable Systems and Methods of Certifying Systems

As part of the seminar a working group discussed methods of developing and certifying systems. This section summarizes their findings.

### 4.6.1 Need

What is happening in today's world or in the future that is driving the need for change?

We are wanting to build increasingly complex and interconnected systems (and systems-of-systems) and are using them in contexts that previously would be considered untenable or unsuitable for software. Our aspirations and reach, as engineers, is growing. The only real tools that we as software engineers have to work with are abstraction and decomposition, though it is questionable whether such reductionist approaches will apply to ultra-large-scale and complex systems. We must fundamentally use models (whether formal or structured) to manage growing complexity, and certification processes must reflect this reality, as well as the new forms of analysis that are available and go beyond what testing can feasibly achieve.

What would we like to be able to do that we can't do now?

- We would like to front-load our analysis and not have to wait until we have a prototype or completely build a system, which may be too late to realize that we are building a system that cannot be certified.
- Replace parts of testing with more rigorous formal analysis/exhaustive analysis and have the results of exhaustive analysis be considered as valid evidence by certifiers.
- Qualifying advanced model-based development tools (which include non-traditional programming constructs – e.g., rule-based programming). Why do we need to develop and use models (of any kind) to help develop certifiable systems? They are useful in particular for evolution.

### 4.6.2 Barriers

What are the barriers (technological, standards, regulatory policy, political, social) that are hindering an advance toward a solution?

Regulatory policy is conservative. Standards evolve slowly and do not necessarily reflect what can be achieved with formal methods or model-based design. This may be a positive thing, given that engineers may be concerned with safety, but having ways to evolve standards that can take into account proven forms of analysis and tooling would be beneficial though we can argue about what “proven” means.

A sociological problem associated with formal methods exists. They have been oversold, misused and misunderstood. This has resulted in bad experiences in critical systems development – typically leading to either hiding use of formal methods (“under the hood”) or not using them at all. Certification is costly to begin with – what effect does formal methods or model-based development have on this cost, especially taking into account the cost of adopting formal methods or model-based development. This links back to John McDermid’s talk on ROI. There has been failures in standardization. For example, UML is conceptually a successful standard but implementation wise (e.g., in terms of XMI/export/input) it is a failure. XMI evolved too slowly and tool vendors did not implement it correctly (or added their own “flavour”).

#### 4.6.3 Evidence that success is possible

What elements of potential solutions exist in research, existing standards, existing technology? There are success stories in applying MBD and Formal methods in different application domains such as nuclear, aerospace and medical devices. We hope to summarize these experiences in the post seminar publication.

#### 4.6.4 Envisioned solution

How would the world (or at least the context of certified systems) change if the challenge could be overcome? What form would solution(s) to the challenge take? new technology? new standards? new methodologies / principled approaches?

Best practice guides to using formal methods or model-based development for building certified systems are critical. This clearly needs some success stories and pilot experiments, of which there are many (at least in the academic literature), but these probably need to be reformulated and expressed differently to get the value proposition to industry across more clearly. Standard interfaces that connect current practice (e.g., modeling in CSV) to new practices (e.g., modeling tools) are also needed.

#### 4.6.5 Roadmap to a solution / research agenda

What concrete steps might the community take to move forward on this challenge?

- Developing methods of analysis for model consistency and coverage
- Clarifying the notions of abstraction used in different models and how to communicate abstractions across domains
- Gateways between methods and tools and improved traceability
- Scale: Your method should scale and you need to provide indicators to decide when it won’t
- Education: Being aware of the notion of “logical proof” already at high school
- Assessment examples (rather like Tim Kelly’s presentation comparing standards) that show how specific MBD or formal methods can produce evidence that is “at least as convincing as” what is indicated in various standards. For example, take DO178C MBD annex and actually compare/assess a specific MBD approach against it to see what constraints need to be applied to the approach, and whether the approach can produce sufficiently compelling evidence.
- If you are an academic with a specific MBD/formal technique, working well on large problems, consider how far are you from something that can be certified.
- A “how-to” guide for building a formal method/MBD technique that can be certified/qualified.

## Participants

- Dominique Blouin  
Université de Bretagne Sud, FR
- Darren Cofer  
Rockwell Collins –  
Cedar Rapids, US
- Cyrille Comar  
AdaCore, Paris, FR
- Mirko Conrad  
The MathWorks GmbH –  
Ismaning, DE
- John S. Fitzgerald  
Newcastle University, GB
- Kim R. Fowler  
Kansas State University, US
- Hubert Garavel  
INRIA Rhône-Alpes, FR
- Janusz Górski  
Gdansk Univ. of Technology, PL
- Arie Gurfinkel  
CMU – Pittsburgh, US
- John Hatcliff  
Kansas State University, US
- Mats P. E. Heimdahl  
University of Minnesota, US
- Constance L. Heitmeyer  
Naval Res. – Washington, US
- Michael Holloway  
NASA Langley ASDC –  
Hampton, US
- Jozef Hooman  
Radboud Univ. Nijmegen, NL
- Jérôme Hugues  
ISAE – Toulouse, FR
- Michaela Huhn  
TU Clausthal, DE
- Hardi Hungar  
German Aerospace Center –  
Braunschweig, DE
- Daniel Kästner  
AbsInt – Saarbrücken, DE
- Peter Karpati  
Institute for Energy Technology –  
Halden, NO
- Vikash Katta  
Institute for Energy Technology –  
Halden, NO
- Tim Kelly  
University of York, GB
- Andrew King  
University of Pennsylvania, US
- John C. Knight  
University of Virginia, US
- Brian Larson  
Multitude Corp., US
- Mark Lawford  
McMaster Univ. – Hamilton, CA
- Dominik Mader  
Berner & Mattner Systemtechnik  
– Berlin, DE
- Tom S. Maibaum  
McMaster Univ. – Hamilton, CA
- John McDermid  
University of York, GB
- Dominique Méry  
LORIA – Nancy, FR
- Frank Ortmeier  
Universität Magdeburg, DE
- Richard F. Paige  
University of York, GB
- Andras Pataricza  
Budapest Univ. of Technology &  
Economics, HU
- Jan Philipps  
Validas AG – München, DE
- Robby  
Kansas State University, US
- Julia Rubin  
IBM – Haifa, IL
- John Rushby  
SRI – Menlo Park, US
- Bernhard Schätz  
fortiss GmbH – München, DE
- David von Oheimb  
Siemens AG – München, DE
- Alan Wassying  
McMaster Univ. – Hamilton, CA
- Jens H. Weber  
University of Victoria, CA
- Virginie Wiels  
ONERA – Toulouse, FR

