Software Engineering Institute

ASSIP Study of Real-Time Safety-Critical Embedded Software-Intensive System Engineering Practices

Peter H. Feiler
Dionisio de Niz

February 2008

SPECIAL REPORT
CMU/SEI-2008-SR-001

Dynamic Systems Program

Carnegie Mellon

This report was prepared for the

SEI Administrative Agent
ESC/XPK
5 Eglin Street
Hanscom AFB, MA 01731-2100

The ideas and findings in this report should not be construed as an official DoD position. It is published in the interest of scientific and technical information exchange.

This work is sponsored by the U.S. Department of Defense. The Software Engineering Institute is a federally funded research and development center sponsored by the U.S. Department of Defense.

Table of Contents

Executive Summary

The Army Strategic Software Improvement Program (ASSIP[1]) has tasked the Carnegie Mellon® Software Engineering Institute (SEI) to conduct a study of real-time, safety-critical, embedded (RTSCE) systems issues and develop recommendations for effectively dealing with those issues. This report contains the results of the first phase, an investigation into the current body of knowledge related to RTSCE system and software development, to include practices employed and emerging in the relevant United States and European commercial and government sectors. In a second phase, issues and shortfalls with the current state of RTSCE software acquisition and development practices will be identified. Recommendations for correcting those problems in the weapon systems domain will be made with emphasis on up-front tasks and considerations that will allow acquirers to position their acquisition programs properly from the start.

Engineering of RTSCE systems requires a rigorous system engineering process in order to meet stringent requirements. This has resulted in the establishment of process standards and practice frameworks for safety-critical systems, with the avionics industry being one of the forerunners. In recent years, the industry trend for RTSCE systems shows a shift to an increased dependence on embedded software to provide value-added capabilities and increased use of a common execution platform with commercially available hardware. This shift has resulted in a new set of system failures due to mismatched assumptions between system engineers and software engineers. A more predictable approach is needed to support the integration of software and computer hardware within the overall system. In other words, software engineers must be allowed to participate with system engineers in the decisions for an RTCSE system and be able to quantify the consequences of those decisions throughout the life cycle.

Two trends show that industry recognizes a need for the improvement of engineering practice: first, process frameworks for system engineering and software engineering such as those of the ISO/IEC (International Organization for Standardization/International Electrotechnical Commission) are being harmonized with a common vocabulary to facilitate better interaction between these two communities; second, the need for predictability during system integration and operation has led to the emergence of a model-based engineering (MBE) practice. This practice is supported by modeling language standards for system engineering (e.g., Systems Modeling Language [SysML]) and for embedded software engineering (e.g., the Architecture Analysis and Design Language [AADL] and Modeling and Analysis of Real-Time and Embedded systems [MARTE]) and a range of analysis methods and tools with strong underlying theories and semantics (e.g., model checking and schedulability analysis). Industry has been the driver in their development and initial use. As these MBE technologies are piloted in industry initiatives, experience is gained about their benefits throughout the development life cycle—including system integration, validation, and certification phases—as well as in their impact on acquisition processes. These experiences can be reflected in revisions to process frameworks. Such pilot work also leads to the de-

[1] Acronyms and initialisms used in this report are listed in the Acronyms and Initialisms section.

® Carnegie Mellon is registered in the U.S. Patent and Trademark Office by Carnegie Mellon University.

velopment of product standards in the form of reference architecture embedded software systems and their runtime infrastructure, expressed in industry standard modeling notations with precise semantics to support quantitative analysis and generation from validated models.

There are opportunities for the U.S. Department of Defense (DoD) to participate in industry initiatives in MBE of embedded software-intensive systems to gain experience, encourage contractors to start using MBE, understand the implications on the acquisition process, and invest in the transition of this practice into DoD programs.

Abstract

Modern weapon systems increasingly depend on real-time, safety-critical, embedded (RTSCE) software to achieve their mission objectives. In addition, these systems are experiencing far longer service lives than anticipated at their inception. Army weapon system developers are concerned that this combination of factors renders today's software acquisition and development practices insufficient to address the challenges of these software-intensive systems. To address the concern, the Army Strategic Software Improvement Program tasked the Carnegie Mellon® Software Engineering Institute (SEI) to assess RTSCE software-intensive systems issues and develop recommendations. The findings of phase one of that study are presented in this report: (1) industry is driving the development of tools for model-based engineering to meet the needs of RTSCE system development, and (2) many opportunities exist for the U.S Department of Defense (DoD) to gain experience and advance the transition of these tools into DoD programs.

1 Introduction

Modern weapon systems increasingly depend on large amounts of real-time, safety-critical, embedded (RTSCE) software to achieve their mission objectives. Additionally, systems are experiencing far longer service lives than anticipated at their inception. Army weapon system developers are concerned that this combination of factors renders today's software acquisition and development practices insufficient to address the challenges of RTSCE software-intensive systems.

To address the concern, the Army Strategic Software Improvement Program (ASSIP)[2] has tasked the Carnegie Mellon® Software Engineering Institute (SEI) to conduct a study of RTSCE software-intensive systems issues and develop recommendations for effectively dealing with those issues. In phase one, the SEI has investigated the current body of knowledge related to RTSCE software development, to include practices employed in the relevant U.S. and European commercial and government sectors. The SEI has briefed the Army Advisory Group (AAG) on its findings; this report reflects the content from the slides used in that briefing and the findings of the study.

This report will be the basis for phase two, where issues and shortfalls with the current state of RTSCE system acquisition and development practices will be identified, and recommendations for correcting those problems in the weapon systems domain will be developed, with emphasis on up-front tasks and considerations that will allow acquirers to properly position their acquisition programs from the start.

In Section 2 of this report, we

- summarize existing and emerging practice standards in three categories: process standards, methods and tool standards, and product standards
- illustrate problems arising from the increased role of software in RTSCE systems and the resulting shift for system integration from a system engineering perspective only to one in which software and computer system engineers become equal partners in the process
- focus on model-based engineering (MBE) as an approach to improve predictability in system integration to address mismatched assumptions between various engineering roles (i.e., system engineers, control engineers, and software engineers)
- show that industry has recognized the maturity of model-based development technology and discuss the importance of using an industry standard modeling notation, with the Society of Automotive Engineers Architectural Analysis and Design Language (SAE AADL) proposed as meeting the need

  In particular, we outline the interplay between the AADL and other modeling notations such as the Unified Modeling Language (UML), Systems Modeling Language (SysML), and Modeling

[2] Acronyms and initialisms used in this report are listed in the Acronyms and Initialisms section.

® Carnegie Mellon is registered in the U.S. Patent and Trademark Office by Carnegie Mellon University.

and Analysis of Real-Time Embedded systems (MARTE), providing a summary of current industry initiatives that are implementing method and tool infrastructures and piloting the use of MBE.

For Section 2, we follow a method of showing and discussing slides from the briefing to AAG.

In Section 3, we recap the impact of the advances in method and tool standardization on process standards and on product standards and outline opportunities for the DoD to get involved in the ongoing process of adopting MBE.

2 The Study Findings

Slide

Safety-Critical Real-Time Embedded Systems

- Software-intensive systems
  - Operational software
  - In context of physical system
  - On target computing platform
- With requirements for
  - Dependability
  - Availability
  - Performance
  - Security
  - Safety-criticality
- Types of systems
  - Embedded systems
  - Real-time systems
  - Systems of systems
- Fields of application
  - Aviation
  - Automotive
  - Aerospace
  - Autonomous systems
  - Medical
  - …

Discussion

Embedded systems have safety-criticality and real-time requirements. Embedded systems today often are systems of systems (i.e., an integrated set of embedded system components). For example, the antilock braking system (ABS) and electronic throttle control used to be considered to be embedded systems, but they are now seen as a set of interacting parts in larger systems such as the electronic stability control.

The integration of these systems is increasingly occurring at the software level and, therefore, depends significantly on the application software architecture, the computer hardware, and interfaces with the physical system.

In this report, we first summarize existing practice standards. Then, we examine the problems of software-intensive embedded systems that have caused researchers and industrial practitioners to invest in model-based technology. Finally, we discuss the status of trends toward improving the state of the practice.

Slide

Existing & Emerging Practice Standards

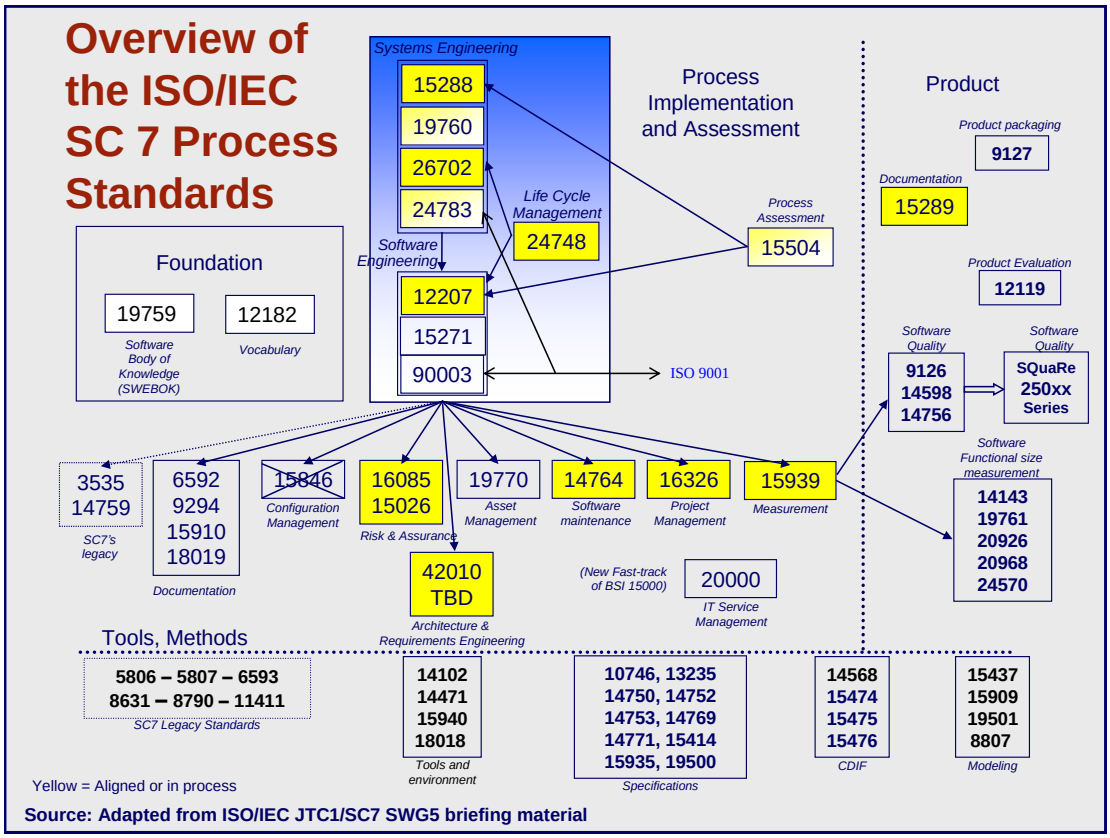
- Process Frameworks
  - Process improvement frameworks
  - Systems and software engineering life cycle processes
  - Embedded Systems: RTCA DO-178B (Safety in Airborne Systems Certification), UK MOD Standard 00-56/3 (Safety Management Requirements for Defence Systems)…
- Methods & Tools Standards and defacto Standards
  - Physical component modeling: Dymola, Simulink
  - System engineering: SysML
  - Computing hardware modeling: VHDL
  - Software: Programming (Ada-Ravenscar, RTJava), Design (HOOD, UML), Architecture (AADL, MARTE)
  - Model interchange: XML & profiles, Model Bus
  - Tool framework: Eclipse plug-in architecture
- Product Standards
  - Physical component interfaces: RS232, PCI,
  - Computing hardware: PPC, RISC, ARM, PCI, CANBus, ARINC 429
  - Infrastructure SW: RT-OS (OSEK, ISO/IEC 9945 [POSIX], ARINC653), OpenCAN, TTP

Discussion

Practice standards for embedded systems fall into three major categories:

1. process standards that focus on management and engineering processes
2. methods and tools standards that focus on modeling notations and languages as well as inter-change standards for describing the system
3. product standards that focus on the components and architecture of the system itself

Slide



Source: Adapted from ISO/IEC JTC1/SC7 SWG5 briefing material

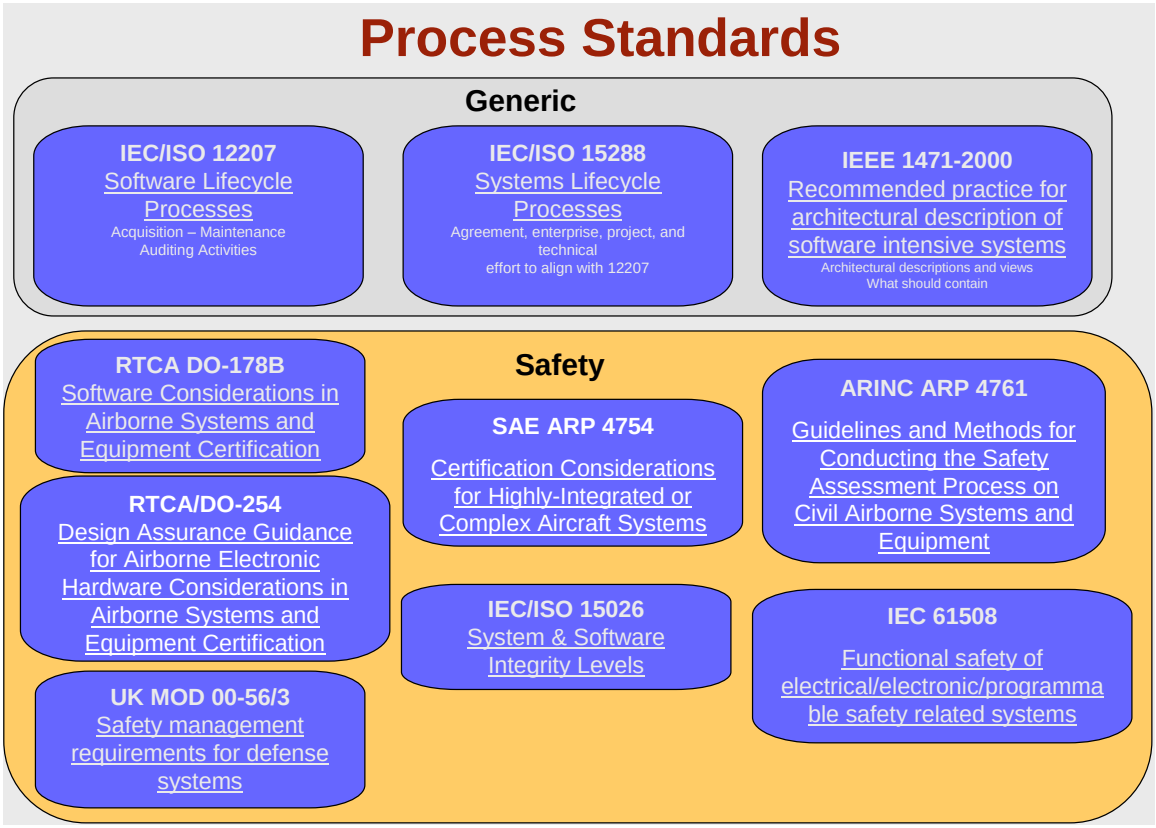
Discussion

A number of process standards have been established with roots in system engineering or software engineering. The International Organization for Standardization/International Electrotechnical Commission (ISO/IEC) is in the process of harmonizing and aligning these two groups of process standards and establishing a common vocabulary.

The process standards fall into three categories:

1. process implementation, management, and assessment
2. product engineering processes
3. tooling processes

Slide



Discussion

In addition to the ISO/IEC process standards on system and software life cycles, the Institute of Electrical and Electronics Engineers (IEEE) has introduced the concept of architecture views and provides a guide for what they should contain.

In avionics, safety-criticality demands and the increasingly software-intensive nature of these systems has led to a number of process standards by a combination of standards organizations. Some standards have been put in place at a national level (e.g., UK MOD Standard 00-56/3) and include guidance to utilize formal methods and safety cases. UK MOD Standard 00-56/3 has recently been revised to be less prescriptive to allow for different formal techniques to be used. Input has been sought to revise RTCA DO-178B[3] [Wikimedia 2005, MOD 2004].

[3] DO-178B, Software Considerations in Airborne Systems and Equipment Certification, is a guidance for software development published by RTCA Incorporated.

Slide

Late Discovery of System Problems

- System integration problems
  - System instability and failures
  - Implicit and mismatched assumptions
  - Shared computing resources
  - Complexity of component interaction
    - Functional
    - Extra-functional
- Current practice
  - Build components first
  - Then integrate and test
- Way forward
  - Analyze system models early and often
  - Evolve components and integrated system

Discussion

As systems have become increasingly software-intensive, new faults and failures are occurring that are not being addressed by traditional fault tolerance techniques. These failures and system instabilities stem from a lack of understanding the impact of choices in the runtime architecture of embedded software systems. With runtime architecture, we refer to the task and communication architecture and its deployment on a distributed computing platform and interface with the physical environment through sensors and actuators.

The cause of these failures is frequently rooted in undocumented and mismatched assumptions, resulting from resource sharing of a shared computing platform, and the increasing complexity of component interactions at the functional and non-functional level. Problems of these types are often discovered during system integration and operational test due to the lack of system-level quantitative analysis of the traditional practice of build first, then integrate and test. The objective of a way forward is to provide predictability earlier in the development life cycle through MBE.

Slide

Impact of Software Integration

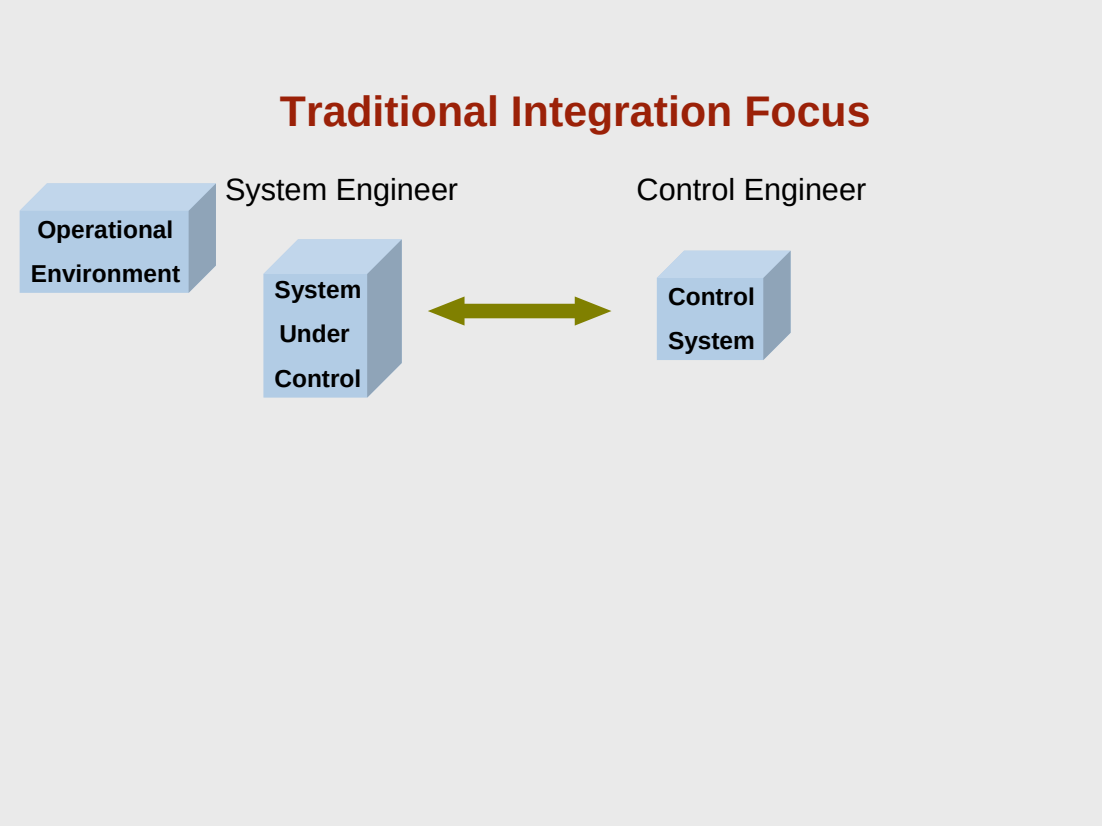
- Mismatched assumptions
  - Units, range, delta, base value (Ariane 4/5)
- False promises of time partitioning
  - DMA impact across partitions (JSF)
- Unmanaged resource sharing
  - Flooding of shared bus (Benz)
- Unexpected latency variation
  - Controller stability
  - Latency increase (Apache)
- Trusting scheduling analysis
  - Detection of priority inversion (Mars Rover)
  - Scheduling results & partitioned architecture

Discussion

This content illustrates different examples of faults due to the fact that the system is implemented in software.

- The Ariane 5 accident was due to assumptions about the physical system that were encoded in legacy software but did not prove to be true when the system was moved from Ariane 4 to Ariane 5 [Wikimedia 2008].
- JSF utilizes a partitioned architecture, as defined by ARINC 653, with the promise of each partition being a virtual processor—giving the impression that it has the processor to itself. However, a direct memory access (DMA) transfer initiated by one partition affects the timing of other partitions.
- Integration of embedded systems on the same computing platform is now common practice in the automotive industry. Daimler had a model configuration where the navigation system could cause the proximity warning system to respond late due to a shared bus whose resource was not managed.
- Migration of embedded applications to different runtime architectures and hardware can result in unexpected latency jitter that can lead to instability of control systems. In the case of F16, the jitter evidenced itself as blurriness of target symbols on the cockpit display.
- In the Mars Pathfinder, priority inversion was detected through system analysis as the root cause of system crashes [Durkin 1998]. Since the deployed system included support for priority ceiling protocols, the problem could be remotely addressed and an otherwise failed mission completed successfully.

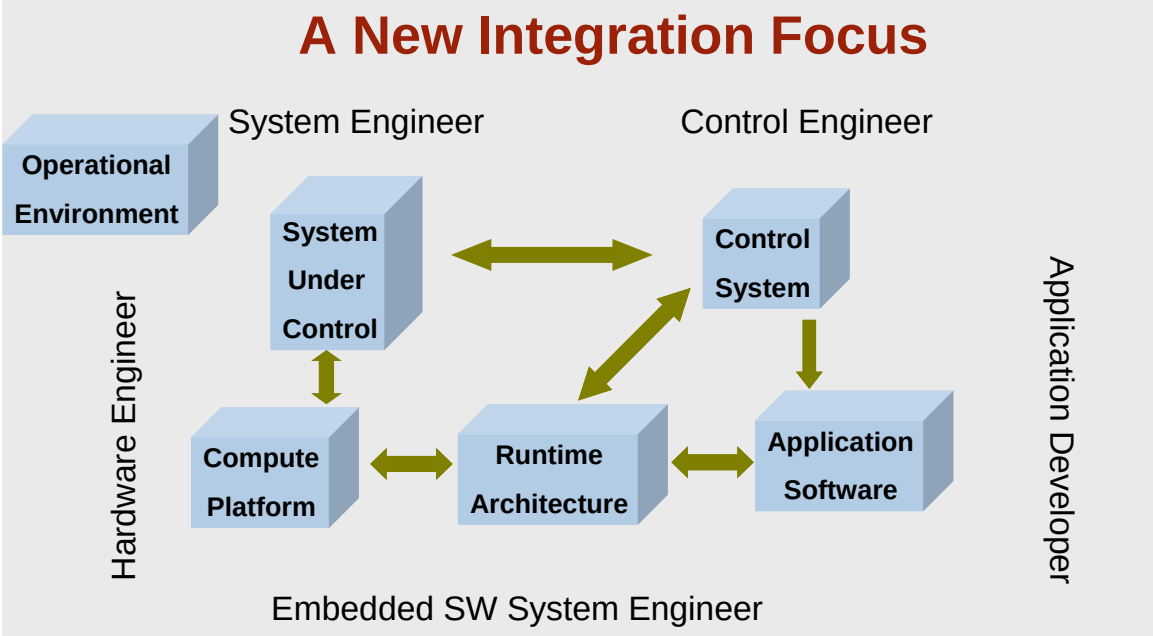
Slide



Discussion

The traditional focus of integration has been in system engineering, centered on the system under control and its operational environment. In the scenario depicted above, the resulting requirements on the control system are refined by control engineers into operational control functionality. The focus is on the physical system and the physical integration from its parts. Software is contained within these individual parts.

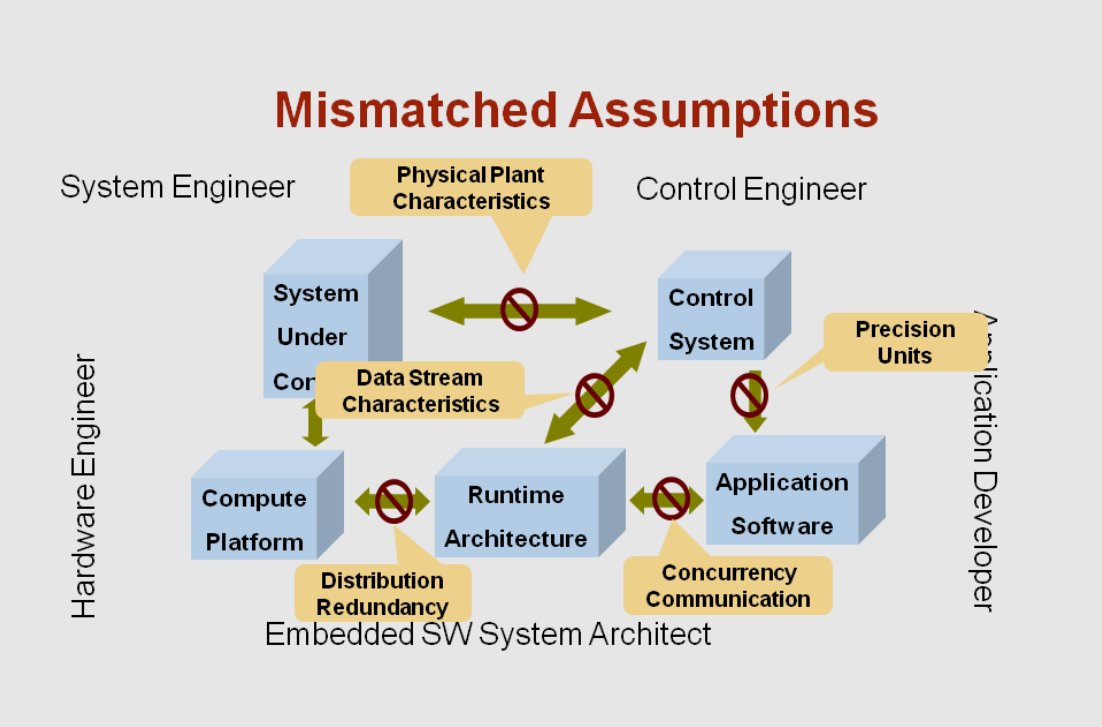
Slide



Discussion

The use of shared computing platforms with multiple interacting applications has led to a new integration focus on the embedded software system, in addition to the integration of the physical system. In this view, the control system, translated into software components, is integrated logically into an interacting set of concurrent tasks in a runtime architecture. This runtime architecture is deployed on a common computing platform, which interfaces with the physical system being controlled. This deployment leads to interactions with additional engineering roles: application developer, hardware engineer, and embedded software system architect.

Slide

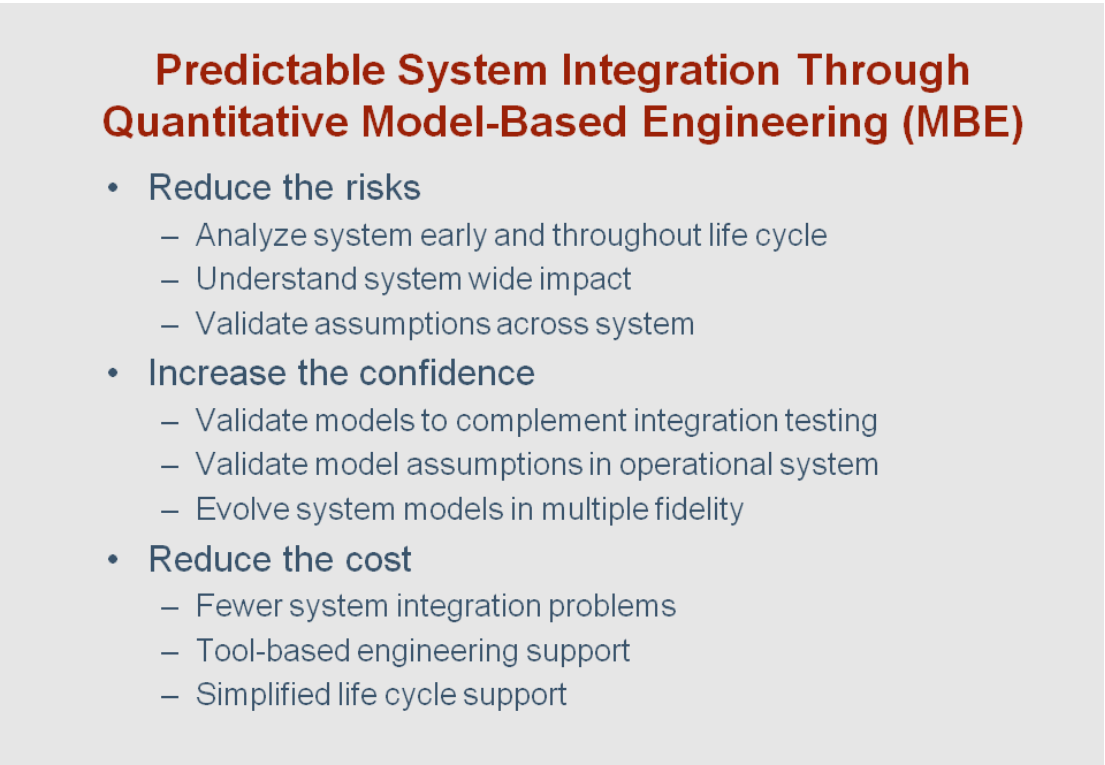


Discussion

Each engineering role (e.g., system engineer or application developer) makes assumptions about its context. These assumptions may not match the assumptions made by other engineering roles.

- System engineers and control engineers may have mismatched assumptions about the physical plant (e.g., the radiation impact on the system under control when in space). In the same way, the translation of control engineers' algorithms into software code by application developers may result in mismatched assumptions about computing precision and measurement units (e.g., the Ariane 5 rounding error due to the 16-bit representation of a variable causing the value to wrap around when the domain value exceeded the 16-bit representation).
- Mapping the application components into a runtime architecture (task and communication topology) may lead to mismatched assumptions about the order of execution due to simultaneous computation (or lack thereof) and timing of task execution and communication (e.g., randomly changing read/write order when communicating through shared variables versus use of double buffering).
- Deployment on the computing platform may lead to mismatched assumptions about redundancy, distribution, and time synchronization (e.g., physical trunk lines of ARPANet [Advanced Research Projects Agency Net] become logical trunk lines when going to fiber optics, reducing physical redundancy to a single instance).
- Assumptions about the data streams being processed by a control system may not be upheld by the chosen task and communication architecture and its binding on the computing hardware (i.e., control engineer's notion of latency does not include latency contributions due to software—preemption, partitioning, protocol overhead, and the like).

Slide



Discussion

Predictability in system integration is achieved through quantitative MBE. Models must be more than pictures: They must have well-defined semantics amenable to computer-based automatic analysis in order to facilitate their quantitative analysis.

Risks are reduced by analyzing models of the embedded software system early and throughout the development life cycle. This continuous modeling can identify and enable discovery of issues from architecture models of the system early in the life cycle, minimizing rework cost.

Confidence in the system is increased by complementing integration testing with quantitative analysis. This can be done in an evolutionary fashion by increasing the fidelity of the models over time as appropriate and by annotating an architecture model with security, performance, and reliability information to drive analysis from a single source model. By generating the analyzed model from the annotated architecture model and validating the generator, we can eliminate the validation step of ensuring that the analysis model reflects the system architecture.

There are several contributors to cost reduction:

- Quantitative analysis allows system integration problems to be addressed earlier in the life cycle.
- Using tools for the analysis is essential, due to the increasing size and complexity of the actual systems.
- Maintaining models throughout the deployment reduces maintenance and evolution costs.

- Improved quantitative modeling reduces recurring costs from overdesign.

Slide

One Aspect of MBE Savings

Company	*Product*	*Tools*	*Specified & Autocoded*	*Benefits Claimed*
Airbus	A340	SCADE With Code Generator	• 70% Fly-by-wire Controls • 70% Automatic Flight Controls • 50% Display Computer • 40% Warning & Maint Computer	• 20X Reduction in Errors • Reduced Time to Market
Eurocopter	EC-155/135 Autopilot	SCADE With Code Generator	• 90 % of Autopilot	• 50% Reduction in Cycle Time
GE & Lockheed Martin	FADEDC Engine Controls	ADI Beacon	• Not Stated	• Reduction in Errors • 50% Reduction in Cycle Time • Decreased Cost
Schneider Electric	Nuclear Power Plant Safety Control	SCADE With Code Generator	• 200,000 SLOC Auto Generated from 1,200 Design Views	• 8X Reduction in Errors while Complexity Increased 4x
US Spaceware	DCX Rocket	MATRIXx	• Not Stated	• 50-75% Reduction in Cost • Reduced Schedule & Risk
PSA	Electrical Management System	SCADE With Code Generator	• 50% SLOC Auto Generated	• 60% Reduction in Cycle Time • 5X Reduction in Errors
CSEE Transport	Subway Signaling System	SCADE With Code Generator	• 80,000 C SLOC Auto Generated	• Improved Productivity from 20 to 300 SLOC/day
Honeywell Commercial Aviation Systems	Primus Epic Flight Control System	MATLAB Simulink	• 60% Automatic Flight Controls	• 5X Increase in Productivity • No Coding Errors • Received FAA Certification

Discussion

Quantitative analysis through MBE is becoming a reality, and industry has recognized its potential. The table above lists a number of projects where behavior of systems is modeled and validated and application code is generated from the validated model. The results are major reductions in errors due to early detection, as well as increases in code productivity due to auto-generation from models.[4]

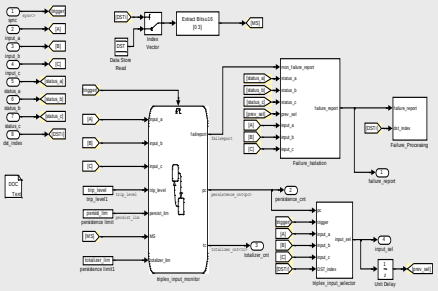
[4] The content shown in the slide on this page is adapted from a Rockwell Collins presentation [Whelan 2007].

Slide

Subtle Errors – LM Aero UAV Sensor Voting

OFP Triplex Voter

- 96 Simulink Subsystems
- 3 Stateflow Diagrams
- $6x10^{13}$ Reachable States



Formal Verification

- 25 Informal Requirements
- 57 Formal Properties
- 40 Minutes to Analyze

10^{100+} reachable states in Rockwell examples

Resulting In

- 24 Counterexamples
- 10 Design Modifications
- Several Requirements Clarifications

Formal verification has found subtle errors that would likely be missed by traditional testing.

- Lockheed Martin

Discussion

Although model checking techniques do not demonstrate the absence of errors, they do discover subtle errors due to increasing complexity and state space explosion that are impossible to find through human examination and testing approaches. Some aspects of model checking technology have been demonstrated to be scalable for realistic systems.[5]

[5] The content shown in the slide on this page is adapted from a Rockwell Collins presentation [Whelan 2007].

Slide

Does Model-Based Development Scale?

Airbus A380

Length	239 ft 6 in
Wingspan	261 ft 10 in
Maximum Takeoff Weight	1,235,000 lbs
Passengers	Up to 840
Range	9,383 miles

Systems Developed Using MBD

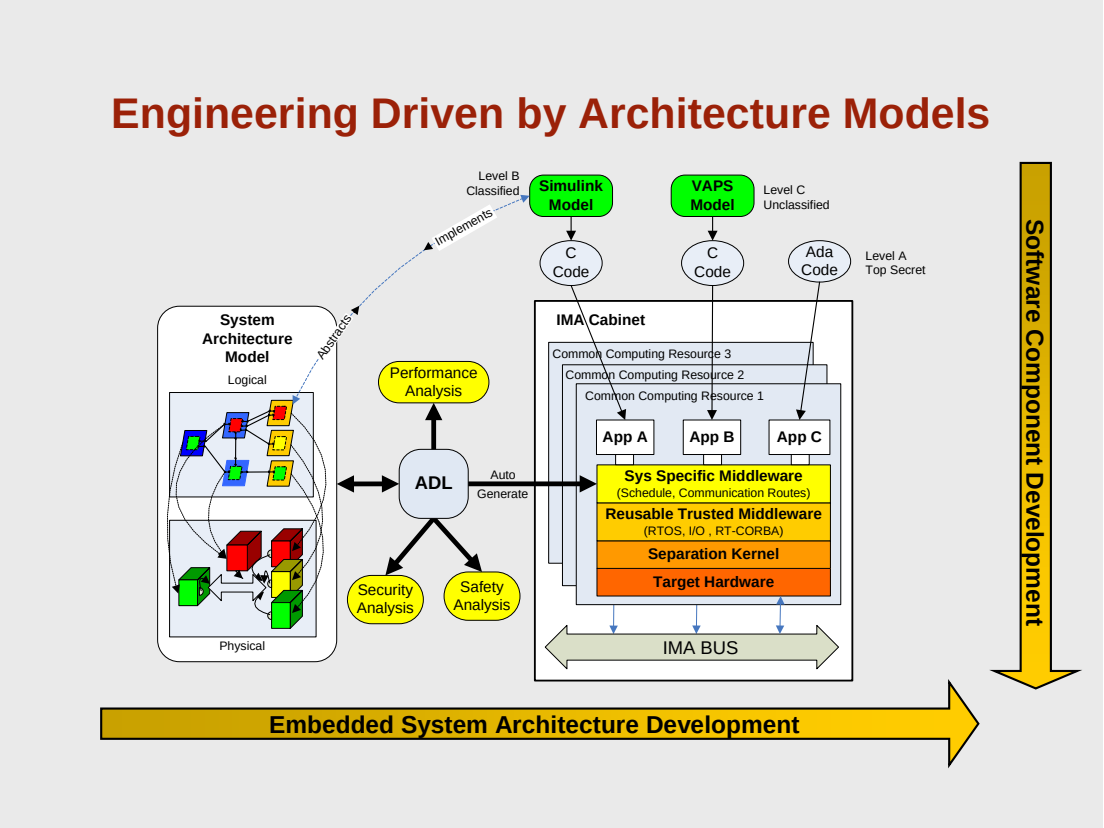
- Flight Control
- Auto Pilot
- Fight Warning
- Cockpit Display
- Fuel Management
- Landing Gear
- Braking
- Steering
- Anti-Icing
- Electrical Load Management

Discussion

Many avionics systems are in operation in which subsystems have been developed through the use of quantitative model-based development (MBD). The number of subsystems developed with this approach continues to increase under the integrated modular avionics (IMA) approach.[6]

[6] The content shown in the slide on this page is adapted from a Rockwell Collins presentation [Whelan 2007].

Slide

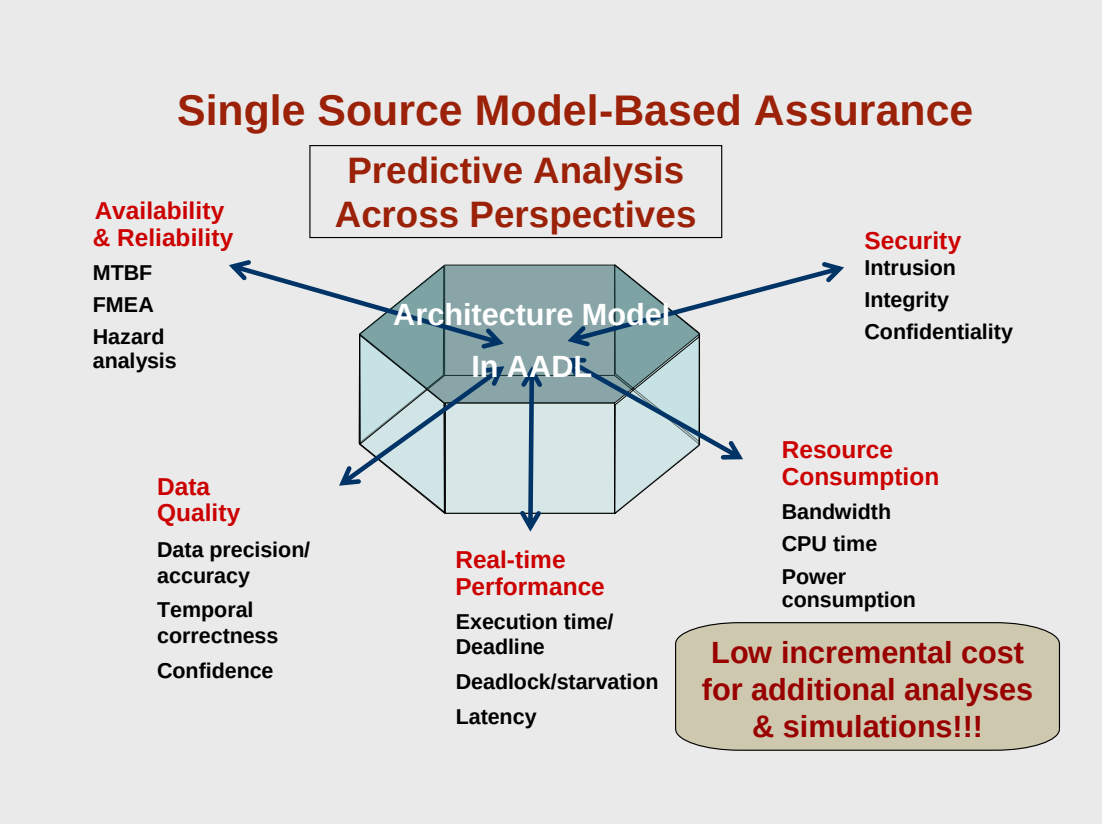


Discussion

MBE triggers changes in the development process. Modeling of individual components occurs through modeling techniques, such as the Mathworks Simulink® platform for control systems. Industry complements the modeling of components with the architectural modeling of the embedded software system through using an architecture description language (ADL) and by driving system-level analysis from the ADL model.[7]

[7] The content shown in the slide on this page is adapted from a Rockwell Collins presentation [Whelan 2007].

Slide



Discussion

By annotating an architecture model with information relevant to each analysis, we have reduced the number of models that have to be validated against the actual architecture. The annotations can be added incrementally to an existing architecture model; as a result, the cost of additional analysis perspectives is minimal. This approach also permits the impact of one perspective on other perspectives to be determined through revalidation of the analyses.

Slide

Benefits of Industry Standard Notation for Embedded Systems Modeling

- Common modeling notation across organizations
- Annotated single source architecture model
- Interchange & integration of architecture models
- Tool interoperability & integrated engineering environments
- Leveraged technology & training investment

Discussion

The benefits of using an industry standard notation for embedded system modeling include

- a common modeling notation across organizations

  This common notation facilitates the reuse of models across projects, improves subcontractor management through ease of model integration, and eases procurement through delivery of analyzable models and analysis results.

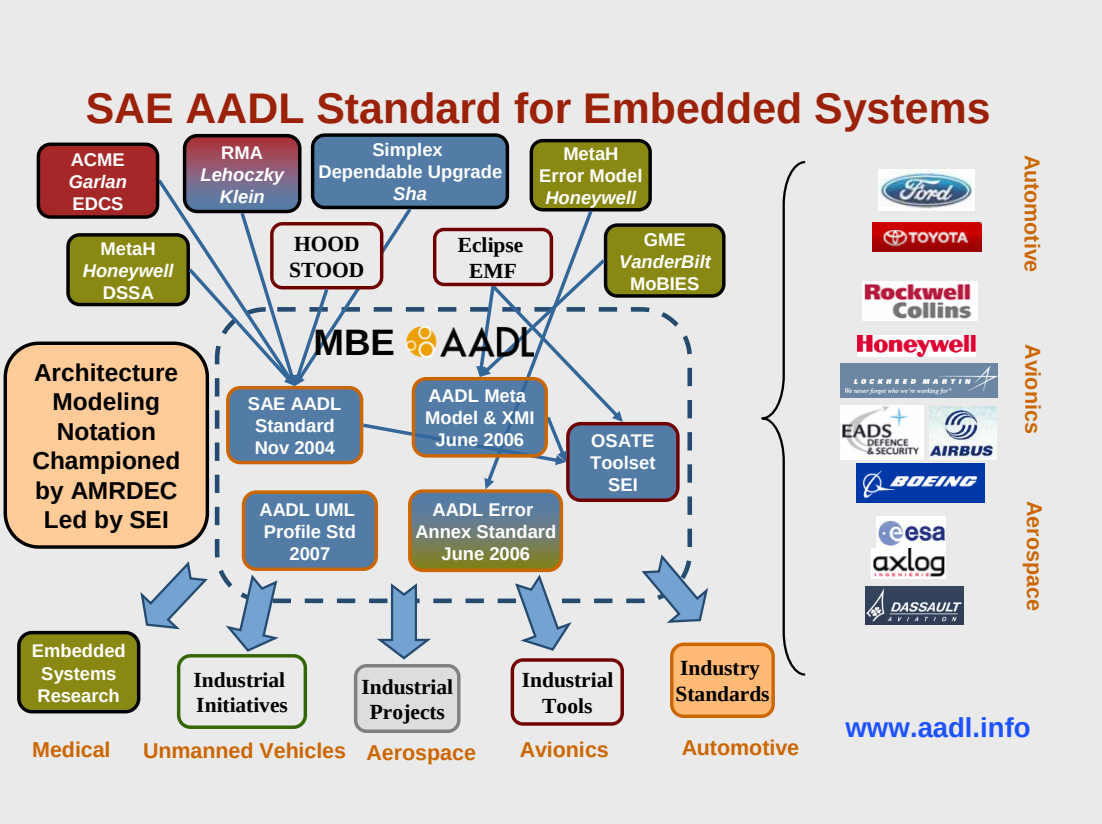
- a single architecture model augmented with analysis properties

  An annotated model reduces validation steps in the development process.

- interchange and integration of architecture models
- tool interoperability and extensible engineering
- leveraged technology and training investment

The promise of these benefits encourages an entire industry to invest in a technology and leverage that tool and training investment.

Slide



Discussion

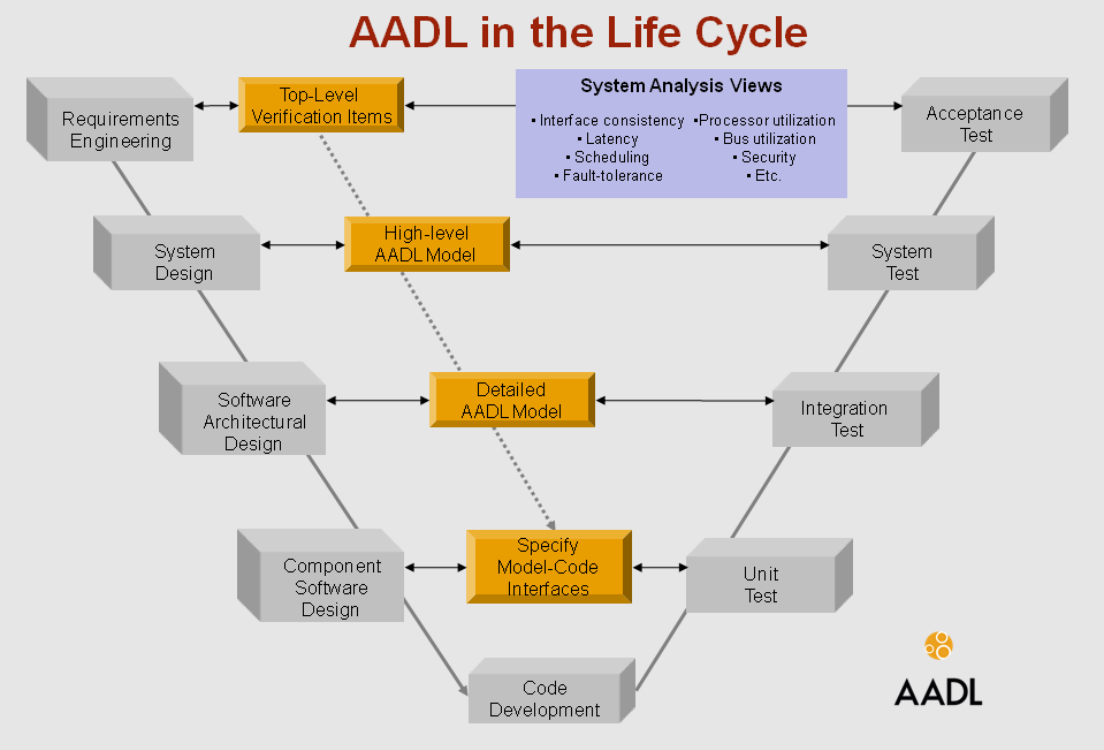
The AADL is an international industry standard, championed by Bruce Lewis (U.S. Army AMRDEC[8]), with technical lead by Peter Feiler (SEI). The AADL focuses on modeling the runtime architecture of embedded software systems (i.e., the task and communication topology, timing and modal behavior, the computing platform and the deployment of the embedded software on this platform, and the physical environment that the embedded system interacts with).

This standard includes industrial technologies; indeed, its development has been driven by industry interests. It also incorporates a number of architecture modeling technologies funded by the Defense Advanced Research Projects Agency (DARPA). The standard was published by the SAE in November 2004 [SAE AS5506 2004, SAE AS5506/1 2006].

Companies in the domains of avionics, aerospace, automotive, unmanned systems, and medical devices are performing pilot projects with this technology. The SAE AADL has also attracted interest from the research community as a research transition platform (i.e., research using AADL becomes quickly accessible to an industry already using AADL) [Lewis 2007, Lewis 2008].

[8] AMRDEC is the Aviation and Missile Research, Development, and Engineering Center.

Slide



Discussion

During the requirements phase, requirements and assumptions about the embedded system can be captured in a simple, high-level parts model. Resource budgets, security, and safety-criticality requirements can be associated with this model and validated for consistency.

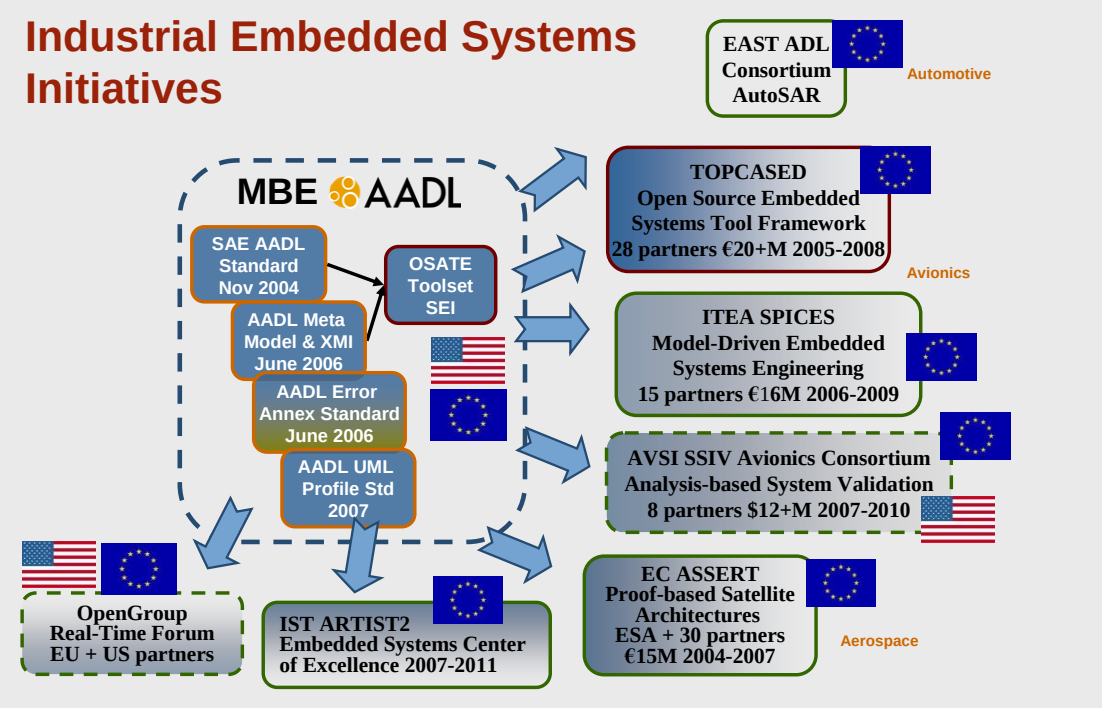
Later, during system design, the computing platform, the interface to the physical environment, and the key embedded software components can be identified. In this phase, decisions regarding the embedded system architecture approach may be captured in a model and analyzed.

During software architecture design, the task and communication architecture of the embedded software system can be refined, and decisions regarding its deployment on a particular computing platform configuration can be analyzed. The consequences of these decisions on the performance, safety-criticality, security, and dependability can be analyzed and validated against requirements.

Software components can be designed and coded in the context of a set of functional and non-functional specifications that have been recorded in embedded system architecture models. They can be validated repeatedly during unit test against test measurements.

Using analysis to validate a set of evolving system models has several positive consequences on the test and integration phases: (1) potential problems can be discovered early in the development life cycle, (2) testing can be focused on validating analysis assumptions in the context of the implemented system, and (3) models can become a source for driving the generation of tests.

Slide



Discussion

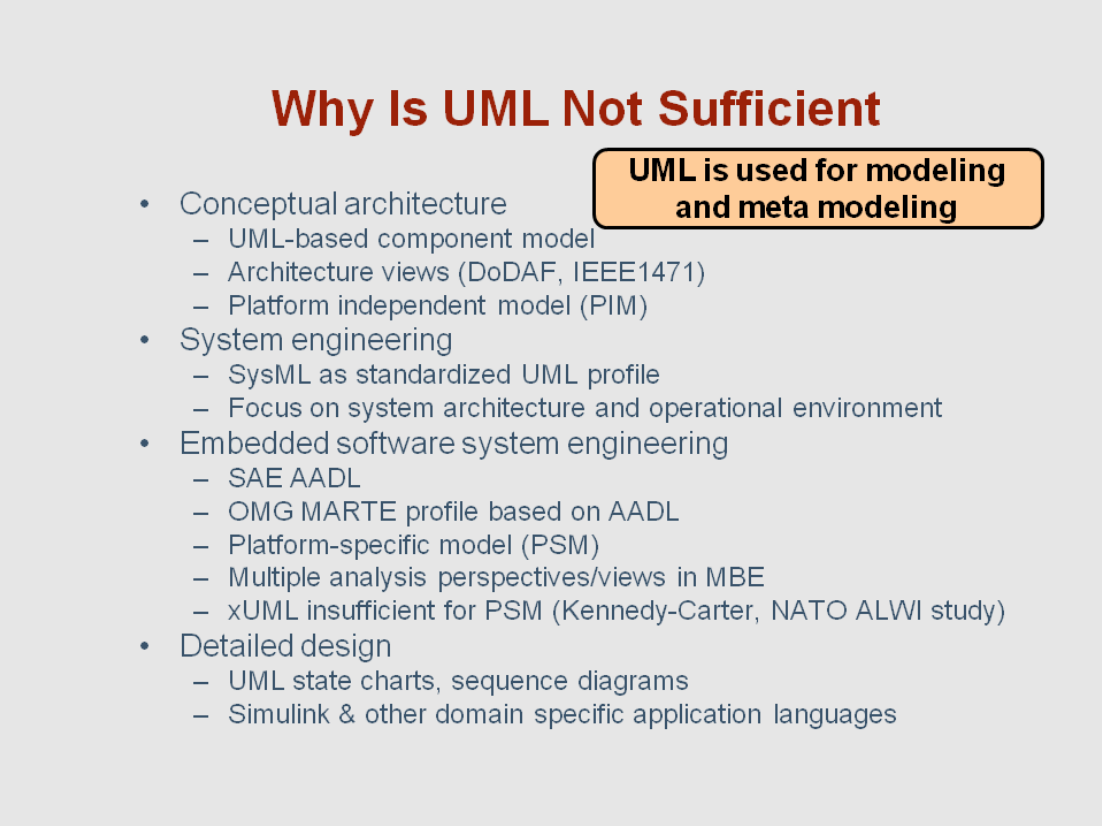
The AADL is being applied in a number of industry initiatives:

- ASSERT (2004-2007, €15M[9]): The European Commission (EC) funded the ASSERT (Automated System and Software Engineering for Real-Time applications) project. ASSERT investigates modeling and analysis of two satellite family architectures to reduce system costs. The European Space Agency leads the project, which has 30 partners.
- TOPCASED (2005-2008, €20M): TOPCASED (Toolkit in OPen source for Critical Applications & SystEms Development), led by Airbus Industries, focuses on an open source tool framework for embedded software systems development. The Open Source AADL Tool Environment (OSATE), which has been provided to the community by the SEI, has become part of the TOPCASED tool suite.
- SPICES (2006-2009, €16M): The SPICES (Support for Predictable Integration of mission Critical Embedded Systems) initiative focuses on methodological issues through the use of AADL in embedded systems, such as the support for the CORBA Component Model (CCM) and automatic generation of runtime system code from AADL models.
- AVSI SSIV (2008-2011, $40M): The Aerospace Vehicle Systems Institute (AVSI) System and Software Integration Verification (SSIV) initiative was to begin in January 2008.
- IST ARTIST2: An Embedded Systems Center of Excellence active in Europe to support interchange between researchers and industry to identify technical challenges.

[9] At the time of the publication of this report, one Euro (€) was equal to about 1.5 U.S. dollars.

- Open Group: This organization fosters the dissemination of technology insights to the practitioner community. The Open Group Real-Time Forum has shown interest in the dissemination of AADL-related insights and has hosted four one-day sessions on AADL in 2007.

Slide

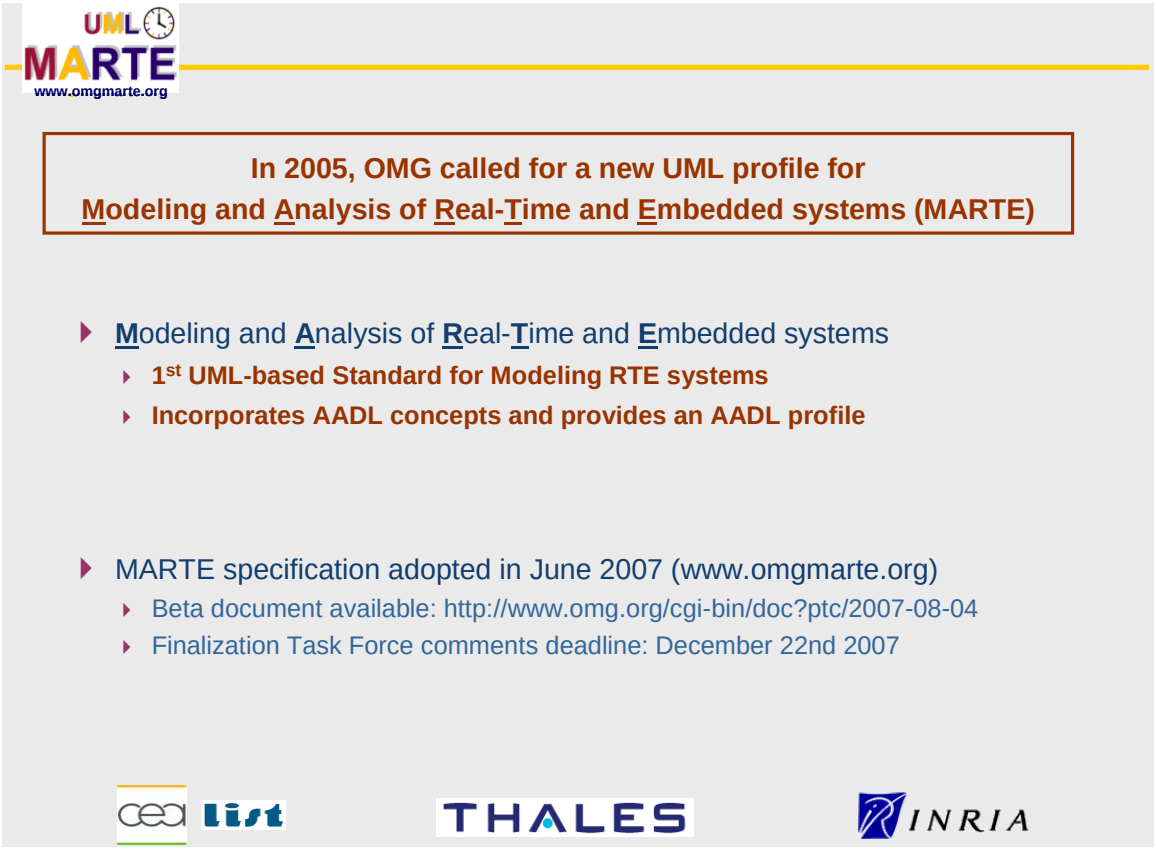


Discussion

UML, an OMG technology, is a popular, graphical notation for component-based modeling and detailed design modeling. It is useful in capturing conceptual architectures in the form of platform-independent models (PIM) of model-driven architecture (MDA), but has shown limitations in representing the platform-specific model (PSM) [Feiler 2007].

Engineers have recognized the need for modeling concepts specific to system engineering and embedded software engineering. Their understanding has resulted in the development of a SysML profile of UML for system engineering and the development of SAE AADL for embedded software system modeling. To support quantitative analysis and model validation, the SysML and AADL modeling notations provide more precise semantics for the language concepts than UML.

Slide



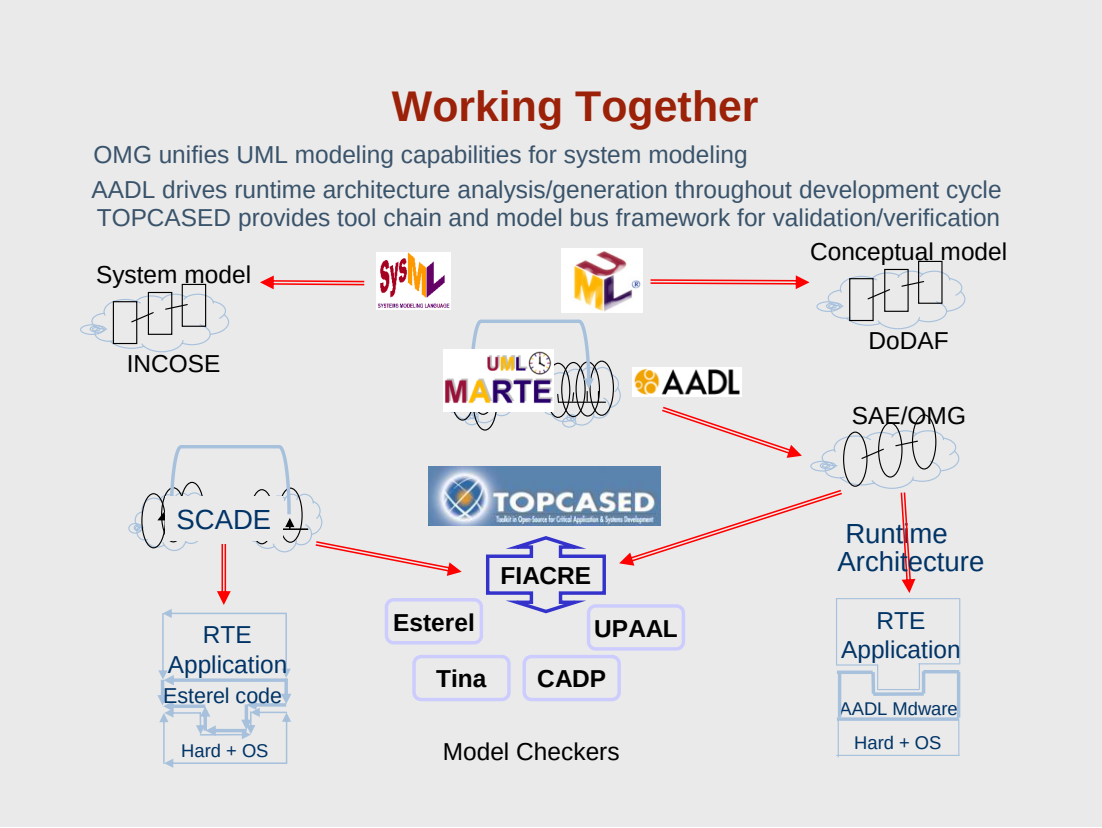
Discussion

Many companies that are using UML see its limitations for PSM. They realize that they are introducing additional but non-standard concepts by profiling stereotypes on top of UML—essentially the same concepts provided by the AADL.

Recognizing the need for better support of modeling for embedded software-intensive systems, the OMG initiated development of the MARTE profile. The OMG and the AADL standards groups are cooperating in the development of that profile.

The MARTE profile provides a set of concepts for real-time and embedded software system modeling on top of UML. It is aligned with SysML concepts and includes an AADL profile [AADL MARTE]. AADL is the standard ADL for embedded software systems; MARTE can be used as a framework to accommodate ADLs in other application domains as well.

Slide



Discussion

The cooperation between AADL and OMG in MARTE allows technology standardization efforts to build on each other, rather than compete, and to leverage other modeling capabilities in the development process. For example, UML may be used when developing models of the conceptual architecture —with multiple views such as those called for in the U.S. Department of Defense Architectural Framework (DoDAF). Also, SysML, a UML extension for system engineering, can be used to capture the system architecture and its engineering concerns.

The AADL and MARTE focus is on the perspective of the embedded software system engineer, whose main responsibility is the integration of application software components on a computing platform. The integrated components need to meet the non-functional requirements of the system as a whole. The TOPCASED initiative defines an interchange notation for model checking representations [Berthomieu 2008]. This notation allows multiple modeling languages to be the source of behavioral models and multiple model checking tools to be applied to them. Similarly, standardization efforts exist for other representations for quantitative analysis, such as the ISO standardization of the Petri net modeling language representation.

Slide

AVSI System and Software Integration Verification

- Overall Concept of Operations
  - Design & production based on early and continuous integration (virtual => physical)
    - Integrate, then Build
- Objective
  - Shift architecting, design, and production activities to explicitly address integration issues early, reducing program execution risks, cycle time and cost
- Approach
  - Adopt/develop “integration-based” software and system development processes with emphasis on integrating Component-Based, Model-Based and Proof-Based Development

Discussion

The AVSI consortium of avionics and aerospace companies is intended to carry out precompetitive research and technology development. The SSIV[10] initiative will perform pilot projects to apply quantitative MBE tools and techniques in a full-scale system design. These pilots will validate the scalability of the technology and address possible changes to procurement, development, validation, and certification processes. AADL has been chosen as a key technology for the first pilot.

The project was in the planning phase in 2007, with participation by Bruce Lewis and Peter Feiler.

[10] The SSIV project name is being changed to System Architecture Virtual Integration (SAVI).

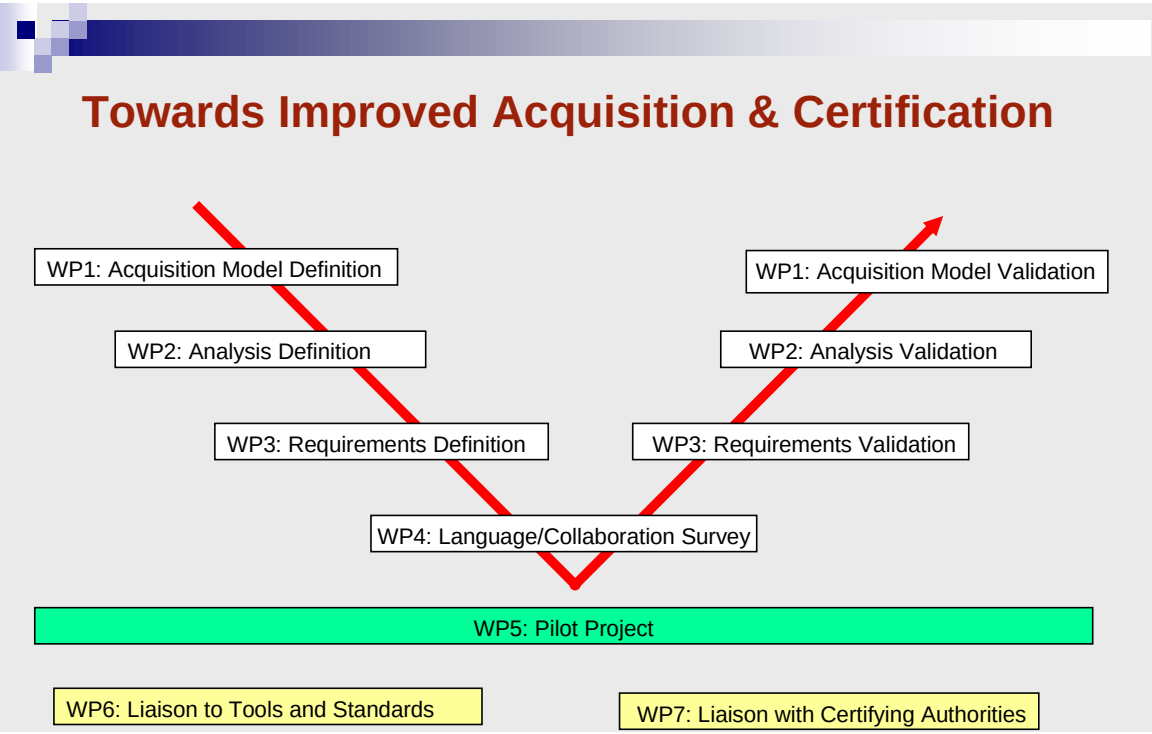
Slide

Discussion

Current membership in the AVSI and SSIV suggests the strong representation of avionics and electronic systems. Government agencies (e.g., Federal Aviation Administration [FAA] and National Aeronautics and Space Administration [NASA]) typically join as liaison members, under a change to the original agreement that allows such members to behave as full members with recognition that their interests are public in nature and are not profit-motivated.

Universities, consultants, research institutions, tool vendors, and standards bodies are not mentioned on this slide but are acknowledged by the SSIV team to be necessary to the success of the project.

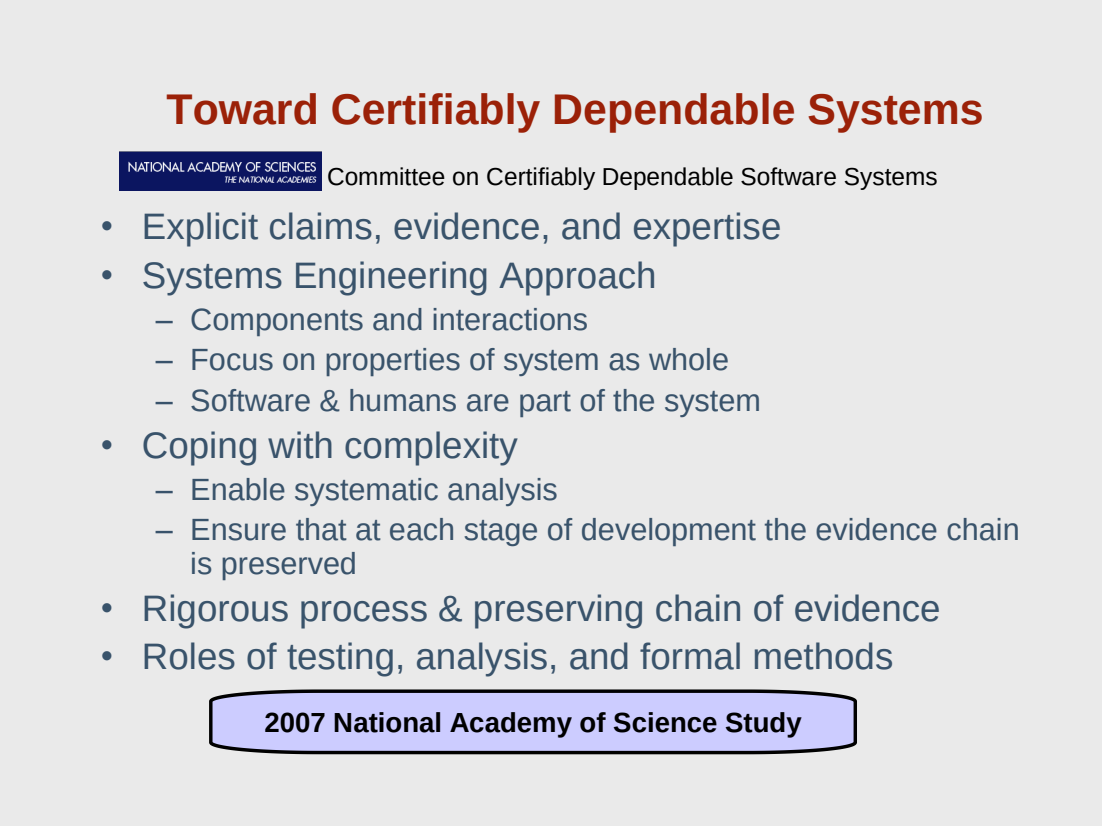
Slide



Discussion

The SSIV project plans include a number of work packages, not only to focus on the execution of the project as a MBE effort but also to build up understanding of changes needed to processes such as requirements and requirements validation, acquisition, validation through analysis, and certification. Furthermore, it is clear that this technology infrastructure should be based on industry standardization as much as possible, with liaisons to the relevant organizations to provide feedback on existing and emerging standards.

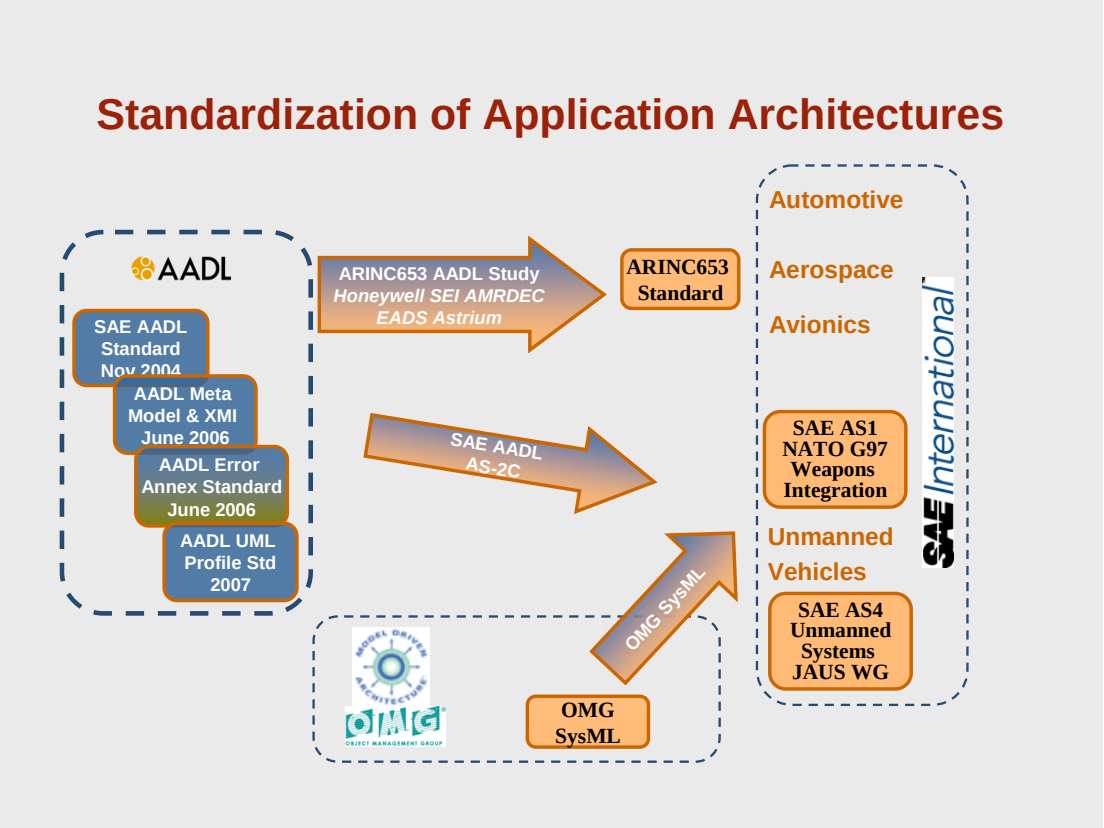
Slide



Discussion

A National Academy of Sciences study in 2007 on certifiably dependable systems concluded, as industry has determined from experience, that it is time to invest in more formal approaches to engineering embedded systems by applying quantitative analysis to models of system architecture with well-defined semantics [Jackson 2007].

Slide



Discussion

Communities that are defining reference architectures and architecture frameworks for their application domain are interested in using the SAE AADL. Several initiatives are utilizing UML, SYSML, and AADL as modeling notations to represent the architecture in an industry standard document.

Three efforts can be highlighted in particular:

1. The SAE AS1 Working Group, with roots in the North Atlantic Treaty Organization (NATO) G97, uses the AADL as a notation to model the embedded systems architecture for weapon systems plug and play; the NATO Air Launched Weapons Integration (ALWI) study combines xUML for platform-independent modeling with AADL for platform-specific modeling.
2. The SAE AS4 Working Group, with roots in JAUS WG (Joint Architecture for Unmanned Systems Working Group), focuses on standardization of unmanned systems architectures and is examining the use of SysML and AADL to capture this architecture [SAE AS1/AS4 2008].
3. ARINC 653 defines a partitioned architecture approach for IMA. Efforts are underway to map ARINC 653 into AADL and use it as a basis to reason about potential issues in migrating existing systems to a partitioned architecture. This may lead to a standardized AADL Annex for ARINC 653 modeling [ARINC 653 2003].

Slide

Discussion

Industry recognizes the need for better predictability of embedded systems, in particular as the integration focus has shifted. No longer exclusively a system engineering activity, integration now also involves embedded software systems.

This recognition has led to standardization efforts in modeling notations and tools, in particular for modeling of the embedded software system architecture. As this technology is piloted, the effect of this shift to quantitative analysis is reflected in revisions to process frameworks, as well as in product standards (in the form of embedded software system architectures and runtime infrastructure).

3 Summary and Conclusion

Engineering of RTSCE systems relies on an established set of process standards and practice frameworks. As dependence on embedded software and common execution platforms has increased, a new set of system failures has emerged. These failures are due to mismatched assumptions between system engineers and software engineers and reveal a need for predictable software integration and migration to new execution platforms. In response, software engineers must become, in effect, "first-class citizens" to participate with system engineers in the decisions for an RTSCE system and quantify the consequences of those decisions.

A need for improvement of the engineering practice has been recognized by industry; two trends have emerged:

- Process frameworks for system engineering and software engineering were originally developed independently by the system engineering community and software engineering community, each using its own vocabulary. To facilitate better interaction between these two communities, process frameworks and standards such as those of ISO/IEC are now being harmonized with a common vocabulary.
- The need for predictability during system integration and deployment has led to the emergence of an MBE practice, known by many names (e.g., MBE, MBSE, MDA, and MIC, among others). A key to MBE is a modeling notation with precise semantics to support quantitative analysis and model validation. The standardization of architecture modeling languages with well-defined semantics for system engineering (e.g., OMG SysML) and embedded software system engineering (e.g., SAE AADL and OMG MARTE) and the maturation of a range of analysis methods and tools with strong underlying theories (e.g., model checking and schedulability analysis) support the application of MBE. Industry has been the driver of the development and initial use of these technologies.

As these MBE technologies are piloted in industry initiatives, the community gains experience in their benefits throughout the life cycle—including system integration, validation, and certification—as well as their impact on acquisition processes. These experiences are reflected in revisions to process frameworks.

Such pilot work also leads to the development of product standards in the form of reference architectures for embedded software systems (e.g., SAE AS4 unmanned systems, SAE AS1 weapons integration) and their runtime infrastructure (e.g., ARINC 653 partitioned systems). These reference architectures and runtime infrastructures can be expressed in industry standard modeling notations with precise semantics, to support quantitative analysis and generation from validated models.

There are opportunities for the DoD to participate in these industry activities in quantitative MBE of embedded software-intensive systems:

- **learn by participation**: There are opportunities to invest and participate in industry pilot programs to apply the emerging MBE practices. In particular, the AVSI SSIV project provides an opportunity. However, there are also opportunities to pilot various ways of introducing MBE into existing DoD projects and programs.
- **encourage contractor use of MBE**: Commercial industry is driving advancement of the MBE practices. DoD contractors are hesitant, however, to incorporate MBE technologies that support quantitative analysis throughout the life cycle into projects until given permission to do so. Such permission allows them to explore how to leverage (and migrate) their projects from UML-based models and other design documentation towards architecture models annotated with relevant functional and non-functional properties. These annotated models can drive quantitative analyses and model validation throughout the life cycle.
- **gain experience**: There are a number of low-cost entry points for introducing MBE practices into projects, including
  - Red teams with experience in MBE may identify root causes of system problems through quantified analysis of architecture models with relevant fidelity. (Red teams provide "critical decision-making expertise during planning and operations" [TRADOC 2005].)
  - In project reviews, experts with experience in MBE may present their understanding of the system architecture through architecture models and analysis of relevant quality attributes.
  - As evaluations of existing and proposed systems are performed (e.g., by using the SEI Architecture Tradeoff Analysis Method® (SEI ATAM® evaluation process), the evidence can be provided in the context of architecture models expressed in SAE AADL, SysML, or MARTE and analyzed through established tool-based analysis techniques. This modeling and analysis can address key concerns about functional and non-functional qualities that have been identified by the stakeholders.
  - Contractors and their suppliers may use an architecture model interchange standard to exchange architecture specification subsystems and systems for virtual system integration into systems and systems of systems, for validation early and throughout the life cycle.
  - As technical risk studies are performed to understand the impact of moving to a new platform, such as a partitioned architecture, model-based analytic frameworks (e.g., for end-to-end latency), can be adapted to account for variations in key non-functional quality measures due to runtime system contributors.
- **technology transition**: As with any new technology inserted into existing practices, MBE requires that some processes be adapted, in particular to enhance the roles of software engineers and architects in integrated system engineering product teams. Also, industry needs to develop a tool and method infrastructure that supports cost-effective model interchange and integration of existing and emerging analysis capabilities into the established engineering environment. In addition,

® Architecture Tradeoff Analysis Method and ATAM are registered in the U.S. Patent and Trademark Office by Carnegie Mellon University.

training must be developed for new software engineering competencies that focus on architecting and integrating embedded software systems through MBE practices. In turn, MBE methods and tools can advance the development of mature product standards in the application domain (e.g., reference architecture and interface standards in support of weapons integration).

- **model-based acquisition process**: MBE employs new methods and tools that change established development and acquisition processes. There is a need to understand the risk of using MBE versus not using MBE. MBE generates new artifacts that can be maintained during the life cycle and delivered as part of the procurement process. Also, system validation through model-based system architecture analysis (virtual integration) early and throughout the life cycle can lead to early discovery of errors, which can result in improved acceptance of testing and certification processes.

From involvement in areas such as those, the DoD would gain experience, encourage contractors to start using MBE, understand the implications on the acquisition process, and invest in the transition of this practice into its programs.

Acronyms and Initialisms

Acronym or Initialism	Description
AADL	**Architecture Analysis and Design Language**
AAG	**Army Advisory Group**
ABS	**Antilock Braking System**
ADL	**Architecture Design Language**
ALWI	**Air Launched Weapons Integration**
AMRDEC	**Aviation and Missile Research, Development, and Engineering Center**
ARPANet	**Advanced Research Projects Agency Net**
ARINC 653	**The avionics application standard software interface**
ARTIST2	**Network of Excellence on Embedded Systems Design**
ASSERT	**Automated System and Software Engineering for Real-Time applications**
ASSIP	**Army Strategic Software Improvement Program**
AVSI	**Aerospace Vehicle Systems Institute**
CCM	**CORBA Component Model**
DARPA	**Defense Advanced Research Projects Agency**
DMA	**Direct Memory Access**
DoDAF	**Department of Defense Architectural Framework**
EC	**European Commission**
EU	**European Union**
FAA	**Federal Aviation Administration**
IEEE	**Institute of Electrical and Electronics Engineers**
IMA	**Integrated Modular Avionics**
ISO/IEC	**International Organization for Standardization/International Electrotechnical Commission**
JAUS WG	**Joint Architecture for Unmanned Systems Working Group**
JSF	**Joint Strike Fighter**
MARTE	**Modeling and Analysis of Real-Time and Embedded systems**
MBD	**Model-Based Development**
MBE	**Model-Based Engineering**
MBSE	**Model-Based Software Engineering & Model-Based System Engineering**

Acronym or Initialism	Description
MDA	**Model Driven Architecture**
MIC	**Model-Integrated Computing**
NASA	**National Aeronautics and Space Administration**
NATO	**North Atlantic Treaty Organization**
OMG	**Object Management Group**
OSATE	**Open Source AADL Tool Environment**
PIM	**Platform Independent Model**
PSM	**Platform Specific Model**
RTSCE	**Real-Time Safety-Critical Embedded**
SAE	**Society of Automotive Engineers**
SPICES	**Support for Predictable Integration of mission Critical Embedded Systems**
SSIV	**System and Software Integration Verification**
SysML	**Systems Modeling Language**
TOPCASED	**Toolkit In OPen source for Critical Applications & SystEms Development**
U.S. DoD	**United States Department of Defense**
UML	**Unified Modeling Language**
xUML	**Executable UML**

References

URLS are valid as of the publication date of this document.

[ARINC 653 2003]
ARINC. *ARINC Specification 653P1-2: 653P1-2 Avionics Application Software Standard Interface, Part 1 - Required Services* (2003). https://www.arinc.com/cf/store/catalog_detail.cfm?item_id=632

[Berthomieu 2008]
Berthomieu, B., et al. "Fiacre: an Intermediate Language for Model Verification in the TOPCASED Environment." *Proceedings of 4th International Congress on Embedded Real-Time Systems (ERTS2008)*. Toulouse, France, January 2008.

[Durkin 1998]
Durkin, Tom. "What the Media Couldn't Tell You About the Mars Pathfinder." *Robot Science & Technology 1*, 1 (1998). http://www.cyberbound.com/docs/1mars.pdf

[Feiler 2007]
Feiler, Peter H, De Niz Dionisio, Raistrick, Chris, Lewis, Bruce. "From PIMS to PSMs," 365−370. *Proceedings of 12th IEEE International Conference on Engineering of Complex Computer Systems (ICECCS07), UML&AADL Workshop*. Auckland, New Zealand, July 2007. IEEE Computer Society Press, 2007.

[Jackson 2007]
Jackson, D., et al., *Software for Dependable Systems: Sufficient Evidence?* National Academies Press, 2007. http://www7.nationalacademies.org/CSTB/pub_dependable.html

[Lewis 2008]
Lewis, Bruce & Feiler, Peter. "Multi-Dimensional Model Based Development for Performance Critical Computer Systems Using the AADL." *Proceedings of 4th International Congress on Embedded Real-Time Systems (ERTS2008)*. Toulouse, France, January 2008.
http://www.aadl.info/documents/ERTS2008_multidimensionalMBE.pdf

[Lewis 2007]
Lewis, Bruce. "System Engineering Approaches for Performance Critical Avionics Embedded Computer Systems Using the Architecture Analysis and Design Language (AADL)." *American Helicopter Society, Forum 63*. Virginia Beach, VA (USA), May 2007 (ISSN 1552-2938).
http://www.aadl.info/documents/AHS_BLewis2007.pdf

[MOD 2004}
UK Ministry of Defence. *UK MOD 00-56/3: Interim Defence Standard 00-56/3: Safety Management Requirements for Defence Systems (December 2004).* http://www.dstan.mod.uk

[SAE AS1/AS4 2008]
Society of Automotive Engineers. *SAE Aerospace Avionics Systems Division*.
www.sae.org/standardsdev/aerospace/aasd.htm (2008)

[SAE AS5506/1 2006]
Society of Automotive Engineers. SAE Standards: AS5506/1, Architecture Analysis and Design Language (AADL) Annex, Volume 1: Graphical AADL Notation, AADL Meta-Model and Interchange Formats, Language Compliance, and Application Program Interface, June 2006. http://www.sae.org/servlets/productDetail?PROD_TYP=STD&PROD_CD=AS5506/1

[SAE AS5506 2004]
Society of Automotive Engineers. SAE Standards: AS5506, Architecture Analysis and Design Language (AADL), November 2004. http://www.sae.org/servlets/productDetail?PROD_TYP=STD&PROD_CD=AS5506

[TRADOC 2005]
U.S. Army Training and Doctrine Command: Office of the Chief of Public Affairs. "Army approves plan to create school for Red Teaming." July 13, 2005. http://www.tradoc.army.mil/pao/tnsarchives/July05/070205.htm

[Whelan 2007]
Whelan, Michael. "Formal Verification of Avionics Software in a Model-Based Development Process." *First International Workshop on Aerospace Software Engineering (AeroSE 07).* http://crisys.cs.umn.edu/icse-workshop/Presentations/(6)AeroSE%20MBD%20and%20FM%20(Whalen).pdf.

[Wikimedia 2008]
Wikimedia Foundation. "Ariane 5 Flight 501." *Wikipedia* (2008). http://en.wikipedia.org/wiki/Ariane_5_Flight_501

[Wikimedia 2005]
Wikimedia Foundation. "DO-178B." *Wikipedia* (2005). http://en.wikipedia.org/wiki/DO-178B

A list of references to standards and related papers mentioned in this report is available upon request from the authors, including:

AADL
A resource on the SAE AADL standard:
http://www.aadl.info
An overview document of the standard:
http://www.aadl.info/documents/AADLLanguageSummary.pdf.
A collection of documents on the use of SAE AADL:
http://www.aadl.info/downloads/AadlStarterKit.zip

AADL MARTE
http://www.omgmarte.org/Documents/tutorial/part8.pdf

ASSERT
http://www.assert-project.net

AVSI SSIV
http://www.aadl.info/documents/AVSI_SSIV_Overview_071121.pdf

IST ARTIST2
Network of Excellence on Embedded Systems Design: http://www.artist-embedded.org

IEC
International Electrotechnical Commission: http://www.iec.ch

ISO
International Standards Organization: http://www.iso.org

MARTE
OMG: http://www.omgmarte.org

SPICES
http://www.spices-itea.org (A project overview presentation can be found at http://www.aadl.info/documents/SPICES_Report_Apr07.pdf.)

SysML
OMG Systems Modeling Language: http://www.sysml.org

UML
OMG Unified Modeling Language: http://www.uml.org

For additional references, write to Peter Feiler (phf@sei.cmu.edu) or Dionisio de Niz (dionisio@sei.cmu.edu).

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. **AGENCY USE ONLY** (Leave Blank)	2. **REPORT DATE** February 2008	3. **REPORT TYPE AND DATES COVERED** Final
4. **TITLE AND SUBTITLE** ASSIP Study of Real-Time Safety-Critical Embedded Software-Intensive System Engineering Practices		5. **FUNDING NUMBERS** FA8721-05-C-0003
6. **AUTHOR(S)** Peter H. Feiler, Dionisio de Niz		
7. **PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)** Software Engineering Institute Carnegie Mellon University Pittsburgh, PA 15213		**8.** **PERFORMING ORGANIZATION REPORT NUMBER** CMU/SEI-2008-SR-001
9. **SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)** HQ ESC/XPK 5 Eglin Street Hanscom AFB, MA 01731-2116		10. **SPONSORING/MONITORING AGENCY REPORT NUMBER**
11. **SUPPLEMENTARY NOTES**		
12A **DISTRIBUTION/AVAILABILITY STATEMENT** Unclassified/Unlimited, DTIC, NTIS		12B **DISTRIBUTION CODE**

13. **ABSTRACT (MAXIMUM 200 WORDS)**

Modern weapon systems increasingly depend on real-time, safety-critical, embedded (RTSCE) software to achieve their mission objectives. In addition, these systems are experiencing far longer service lives than anticipated at their inception. Army weapon system developers are concerned that this combination of factors renders today's software acquisition and development practices insufficient to address the challenges of these software-intensive systems. To address the concern, the Army Strategic Software Improvement Program tasked the Carnegie Mellon Software Engineering Institute (SEI) to assess RTSCE software-intensive systems issues and develop recommendations. The findings of phase one of that study are presented in this report: (1) industry is driving the development of tools for model-based engineering to meet the needs of RTSCE system development, and (2) many opportunities exist for the U.S Department of Defense (DoD) to gain experience and advance the transition of these tools into DoD programs.

14. **SUBJECT TERMS** ASSIP, embedded systems, real-time, software-intensive	15. **NUMBER OF PAGES** 50
16. **PRICE CODE**	

17. **SECURITY CLASSIFICATION OF REPORT** Unclassified	18. **SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	19. **SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	20. **LIMITATION OF ABSTRACT** UL

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89) Prescribed by ANSI Std. Z39-18
298-102