A Four Pillar Improvement Strategy to Qualification of Embedded Software Systems

Peter Feiler

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8702-15-D-0002 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

DM18-0609

Outline

- Challenges in Safety-critical Software-intensive Systems
- Four Pillar Improvement Strategy
- Pillar One
- Pillar Two
- Pillar Three
- Pillar Four

We Rely on Software for Safe Aircraft Operation

Quantas Airbus A330-300 Forced to Landing - 36 Injured

Written by htbw on Oct-7-08 1:48pm

From: **soyawannaknow.blogspot.com**

Thirty-six passengers ar
in a mid-air drama that fo
emergency landing, the
Tuesday.

The terrifying incident sa
mayday call when it sudd
from Singapore to Perth, Qantas said.

Oct. 15 (Bloomberg) -- **Airbus SAS** issued an alert to air
after Australian investigators said a computer fault on a
Ltd. flight switched off the autopilot and generated false
jet to nosedive.

The Airbus A330-300 was cruising at 37,000 feet (11,27
computer fed incorrect information to the flight control s
Australian Transport Safety Bureau said yesterday. Th
650 feet within seconds, slamming passengers and crew
ceiling, before the pilots regained control.

``This appears to be a unique event,'' the bureau said,
Toulouse, France-based Airbus, the world's largest mak
aircraft, issued a telex late yesterday to airlines that fly
fitted with the same air-data computer. The advisory is
minimizing the risk in the unlikely event of a similar occu

Not logged in Talk Contributions Create account Log

WIKIPEDIA
The Free En

Article Talk | Read Edit View history Search

Main page
Contents
Featured co
Current eve
Random art
Donate to W
Wikipedia store

Interaction
Help
About Wikipedia
Community portal
Recent changes
Contact page

Tools
What links here
Related changes

Embedded software systems introduce a new class of problems not addressed by traditional system safety analysis

maker had intentionally programmed turbocharged direct injection (TDI) diesel engines to activate certain emissions controls only during laboratory emissions testing. The programming caused the vehicles' nitrogen oxide (NO_x) output to meet US standards during regulatory testing, but emit up to 40 times more NO_x in real-world driving.[11] Volkswagen put this programming in about eleven million cars worldwide, and in 500,000 in the United States, during model years 2009 through 2015.[12][13][14][15][16]

These findings stemmed from a study on emissions discrepancies between European and US models of vehicles commissioned in 2014 by the International Council on Clean Transportation (ICCT), summing up the data from three different sources on 15 vehicles. Among the research groups was a group of five

2010 VW Golf TDI with defeat device displaying "Clean Diesel" at a US auto show.

Date 2008–2015

FAA says software problem with Boeing 787s could be catastrophic

By **Dan Catchpole**
@dcatchpole

The Federal Aviation Administration says a software problem with Boeing 787 Dreamliners could lead to one of the advanced jetliners losing electrical power in flight, which could lead to loss of control.

- **The Buzz**: Hipster's dilemma
- Boeing & aerospace news
- Aerospace blog

The FAA notified operators of the airplane Friday that if a 787 is powered continuously for 248 days, the plane will automatically shut down its alternating current (AC) electrical power.

Software Problems not just in Aircraft

Expert • Independent • Nonprofit

ConsumerReports.org

This article appeared in May 2010 Consumer Reports Magazine.

Many appliances now rely on electronic controls and operating softw... But it turned out to be a problem for the Kenmore 4027 front-loader, which scored near the bottom in our February 2010 report.

Our tests found that the rinse cycles on some models worked improperly, resulting in an unimpressive cleaning.

When Sears, which sells the washer, saw our February 2010 Ratings (available to subscribers), it worked with LG, which makes the washer, to figure out what was wrong. They quickly determined that a software problem was causing short or missing rinse and wash cycles, affecting wash performance. Sears and LG say they have reprogrammed the software on the models in their warehouses and on about 65 percent of the washers already sold, including the ones we had purchased.

Our retests of the reprogrammed Kenmore 4027 found that the cycles now worked properly, and the machine excelled. It now tops our Ratings (available to subscribers) of more than 50 front-loaders and we've made it a CR Best Buy.

If you own the washer, or a related model such as the Kenmore 4044 or Kenmore Elite 4051 or 4219, you should get a letter from Sears for a free service call. Or you can call 800-733-2299.

May 7, 2010

Lexus GX 460 passes retest; Consumer Reports lifts "Don't Buy" label

Consumer Reports is lifting the Don't Buy: Safety Risk designation from the 2010 Lexus GX 460 SUV after recall work corrected the problem it displayed in one of our emergency handling tests. (See the original report and video: "Don't Buy: Safety Risk--2010 Lexus GX 460.")

We originally experienced the problem in a test that we use to evaluate what's called lift-off oversteer. In this test, as the vehicle is driven through a turn, the driver quickly lifts his foot off the accelerator pedal to see how the vehicle reacts. When we did this with our GX 460, its rear end slid out until the vehicle was almost sideways. Although the GX 460 has electronic stability control, which is designed to prevent a vehicle from sliding, the system wasn't intervening quickly enough to stop the slide. We consider this a safety risk because in a real-world situation this could cause a rear tire to strike a curb or slide off of the pavement, possibly causing the vehicle to roll over. Tall vehicles with a high center of gravity, such as the GX 460, heighten our concern. We are not aware, however, of any reports of injury related to this problem.

Lexus recently duplicated the problem on its own test track and developed a software upgrade for the vehicle's ESC system that would prevent the problem from happening. Dealers received the software fix last week and began notifying GX 460 owners to bring their vehicles in for repair.

We contacted the Lexus dealership from which we had anonymously bought the vehicle and made an appointment to have the recall work performed. The work took about an hour and a half.

Following that, we again put the SUV through our full series of emergency handling tests. This time, the ESC system intervened earlier and its rear did not slide out in the lift-off oversteer test. Instead, the vehicle understeered—or plowed—when it exceeded its limits of traction, which is a more common result and makes the vehicle more predictable and less likely to roll over. Overall, we did not experience any safety concerns with the corrected GX 460 in our handling tests.

How do you upgrade washing machine software?

Internet of Things!

Cost/Capability Limit on Avionics Systems

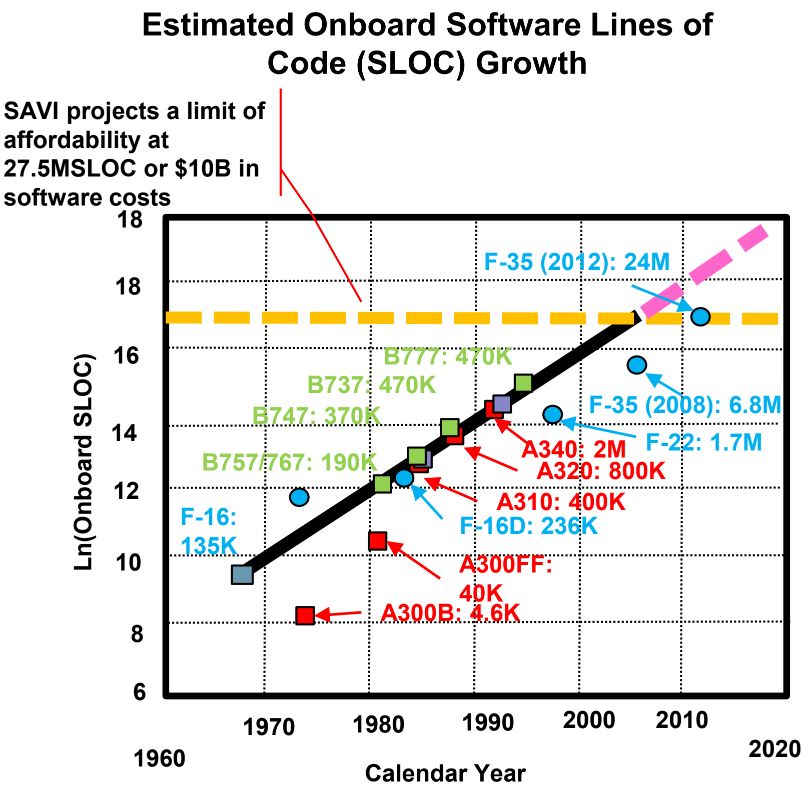
As size and complexity increases so do total development and rework costs.

SAVI predicts that cost growth is unsustainable.

Software as % of total system development

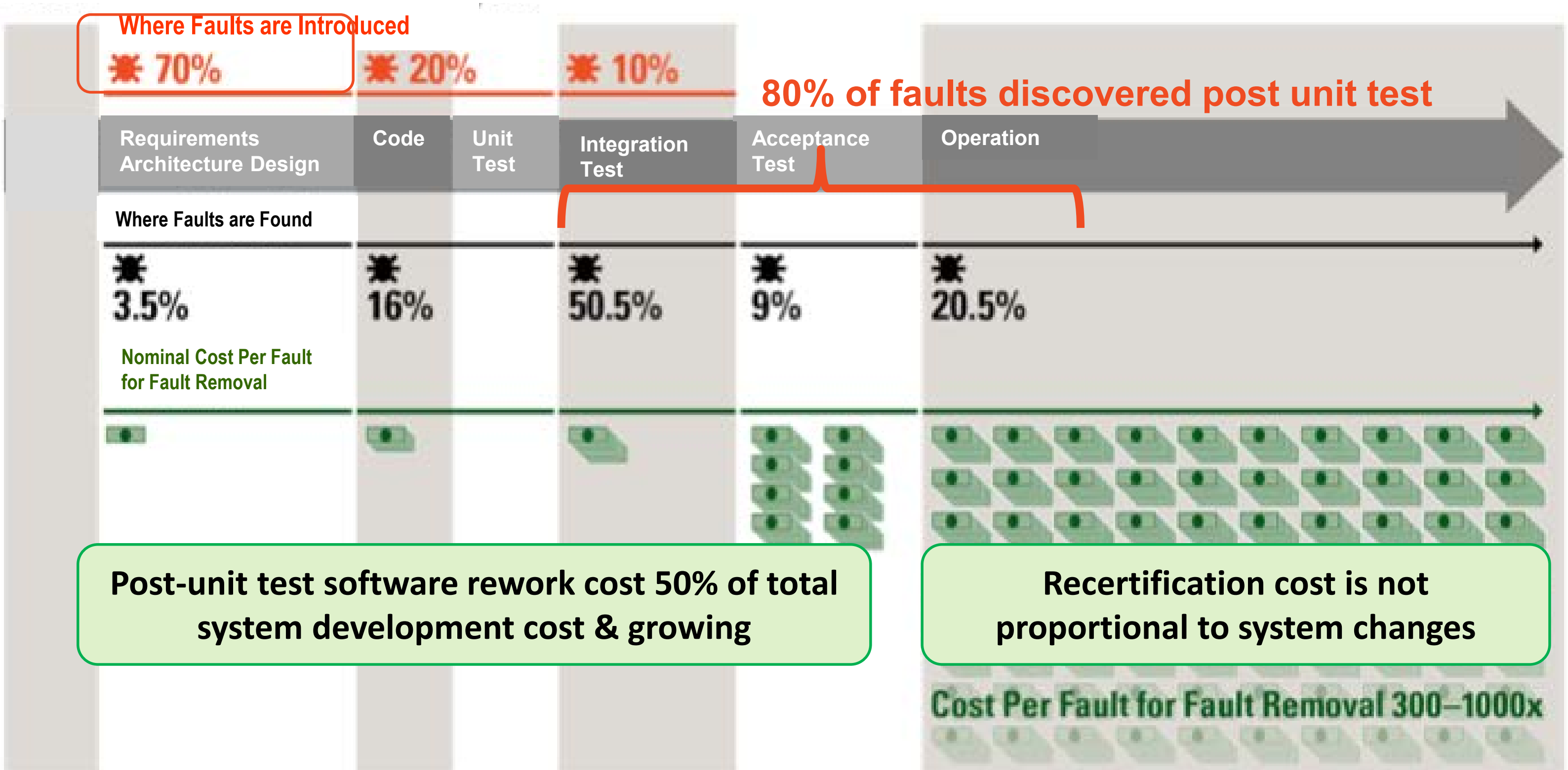
- 45% in 1997
- 66% in 2010
- To exceed 80% by 2020

SAVI System Architecture Virtual Integration — A Commercial Aviation Industry Consortium



Airbus data source: J. P. Potocki De Montalk, "Computer Software in Civil Aircraft," Sixth Annual Conference on Software Assurance (Compass '91), Gaithersburg, MD, June 24-27, 1991
Boeing data source: J. J. Chilenski, 2009

Critical System Assurance Challenges



Sources: *Critical Code*; NIST, NASA, INCOSE, and Aircraft Industry Studies

Current Industry Practice in DO-178B Compliant Requirements Capture

Industry Survey in 2009 FAA Requirements Engineering Study

Notation

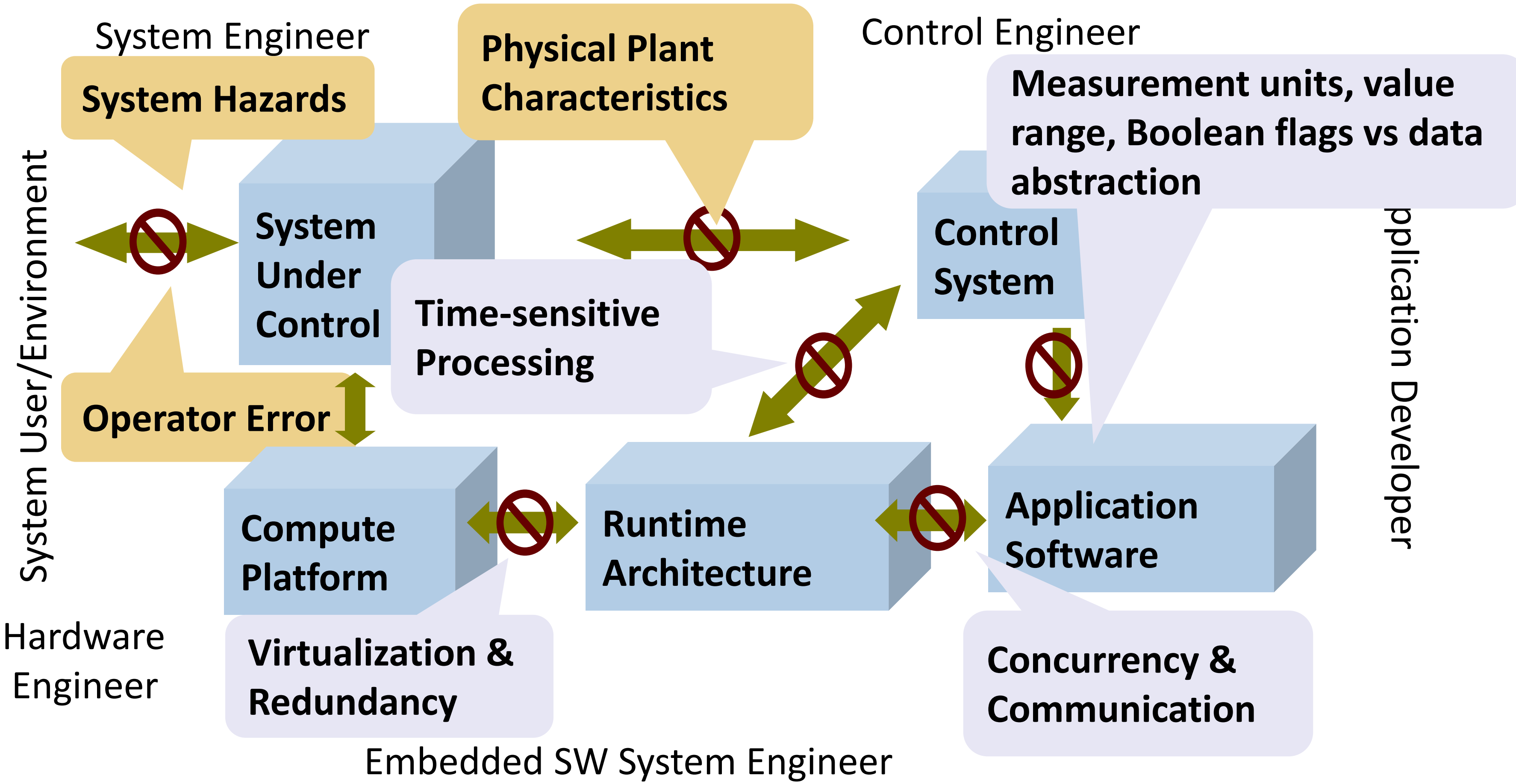
Enter an "x" in every row/column cell that applies	System Requirements	Data Interconnect {ICD}	High-Level Software Requireme	Low-Level Software Requiremer	Hardware Requirements
English Text or Shall Statements	39	27	36	32	29
Tables and Diagrams	31	30	30	19	18
UML Use Cases	1		2	4	
UML Sequence Diagrams			3	6	
UML State Diagrams			1	7	
Executable Models (e.g. Simulink, SCADE Suite, etc.)	7	1	8	8	1
Data Flow Diagrams (e.g. Yourdon)	4		6	9	
[illegible]					
Other (Specify)XML		1			
Operational models or prototypes	1	1			1
UML			1	1	

Need analyzable & executable specifications

Tool

Enter an "x" in every row/column cell that applies	System Requirements	Data Interconnect {ICD}	High-Level Software Requirements	Low-Level Software Requirements	Hardware Requirements
Database (e.g., Microsoft Access)	3	4	3	3	
DOORS	23	13	22	18	12
Rational ROSE®			1	3	
RDD-100®					
Requisite Pro®	5	3	5	4	4
Rhapsody	1				
SCADE Suite	2		3	1	
Simulink	5	1	5	3	1
Slate	1		1	1	
Spreadsheet (e.g., Microsoft Excel)	5	4	5	4	3
Statemate					
Word Processor (e.g., Microsoft Word)	19	20	18	17	16
VAPS™		1	3	3	
Designer's Workbench™			1	1	
Proprietary Database, SCADE like pic tool		1	1		
Interleaf	1	1	1	1	1
BEACON	1	1	1	1	
CaliberRM	1	1	1	1	1
XM:		1			
Wiring diagram		1			1

Mismatched Assumptions and Concepts in Safety-Critical System Interactions



Embedded software system as major source of hazards

Why do system level failures still occur despite fault tolerance techniques being deployed in systems?

Outline

Challenges in Safety-critical Software-intensive Systems

▶ Four Pillar Improvement Strategy

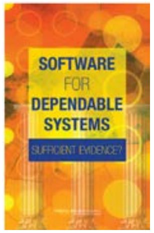
Pillar One

Pillar Two

Pillar Three

Pillar Four

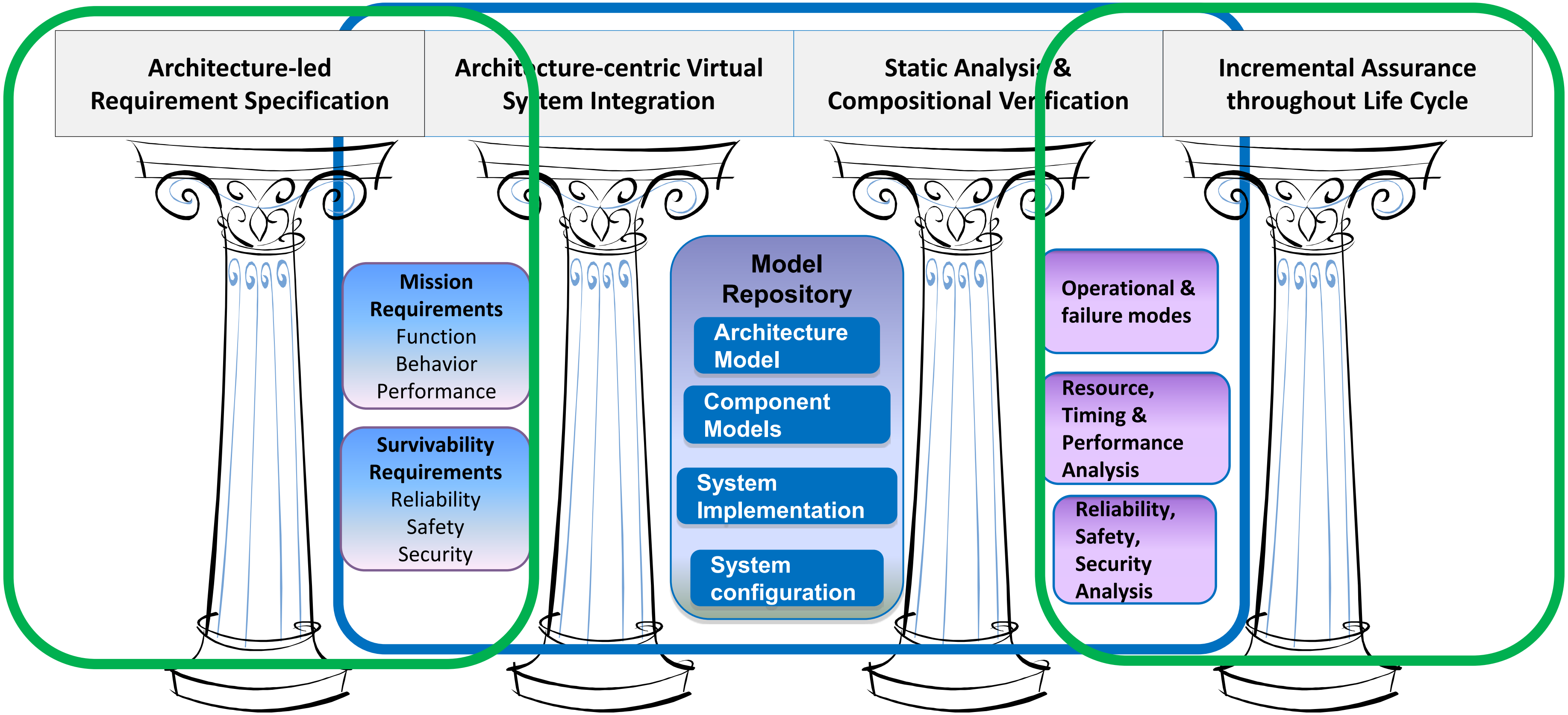
Assurance & Qualification Improvement Strategy



Assurance: Sufficient evidence that a system implementation meets system requirements

2007 National Research Council Study

2010 SEI Study for AMRDEC Aviation Engineering Directorate

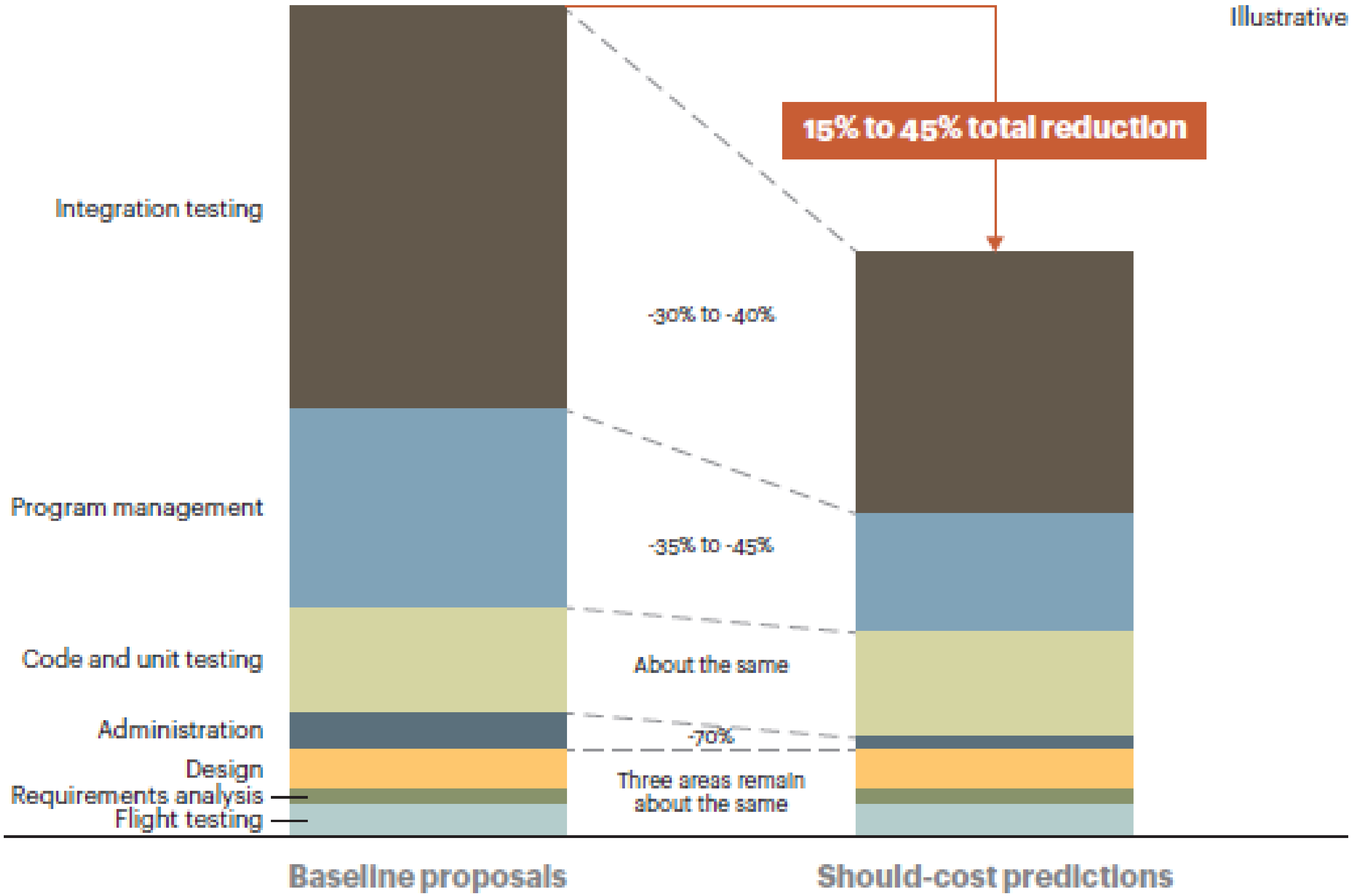


Early Problem Discovery through Virtual System Integration & Analysis

Improved Assurance through Better Requirements & Automated Verification

"Should Cost" Predictions For Avionics

SAVI Return on Investment Study
CMU/SEI-2018-TR-002



ATKearney "Software: The Brains Behind U.S. Defenses Systems"

Outline

- Challenges in Safety-critical Software-intensive Systems
- Four Pillar Improvement Strategy
- Verifiable System Requirement/Specification
- Architecture-Centric Virtual System Integration
- Compositional Verification to Complement Testing
- Incremental System Assurance throughout Development
- Conclusion

Value of Requirement Uncertainty Awareness

Textual requirement quality statistics

- Opportunity for high payoff improvement
- Focus on verifiable requirement specification

Requirements error	%
Incomplete	21%
Missing	33%
Incorrect	24%
Ambiguous	6%
Inconsistent	5%

Experience based uncertainty measures

- 80% of requirement changes from development team
- Requirement uncertainty contributors
  - Volatility, Impact, Precedence, Time criticality
- Awareness of requirement uncertainty reduces requirement changes by 50%
  - Focus on uncertainty areas
  - Engineer for inherent variability

Selection	Weight	Precedence
Low Precedence	9	No experience of concept, or environment. Historically volatile
Medium Precedence	3	Some experience in related environments. Some historic volatility
High Precedence	1	Concept already in service. Low historic volatility

Rolls Royce case study

Requirements & Architecture Design Specification

Example borrowed from M. Whelan

Requirements for a Patient Therapy System

The patient shall never be infused with a single air bubble more than 5ml volume.

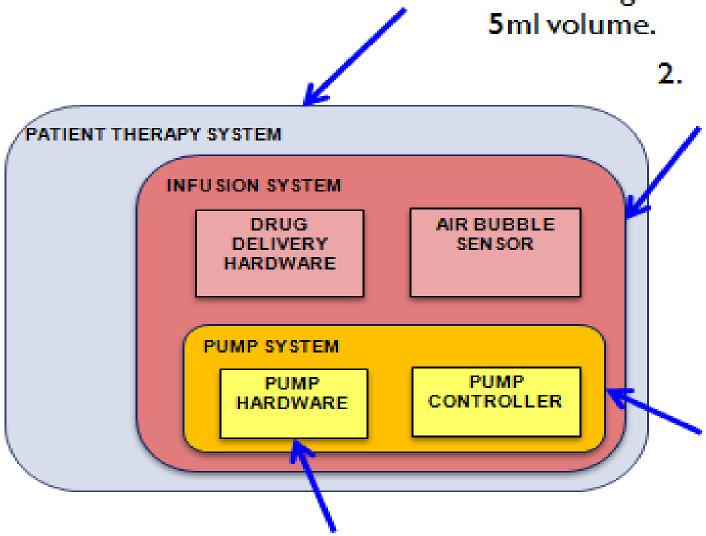
When a single air bubble more than 5ml volume is detected, the **system** shall stop infusion within 0.2 seconds.

When piston stop is received, the **system** shall stop piston movement within 0.01 seconds.

The **system** shall always stop the piston at the bottom or top of the chamber.

Requirements and Design Information

1. The patient shall never be infused with a single air bubble more than 5ml volume.
2. When a single air bubble more than 5ml volume is detected, the **system** shall stop infusion within 0.2 seconds.



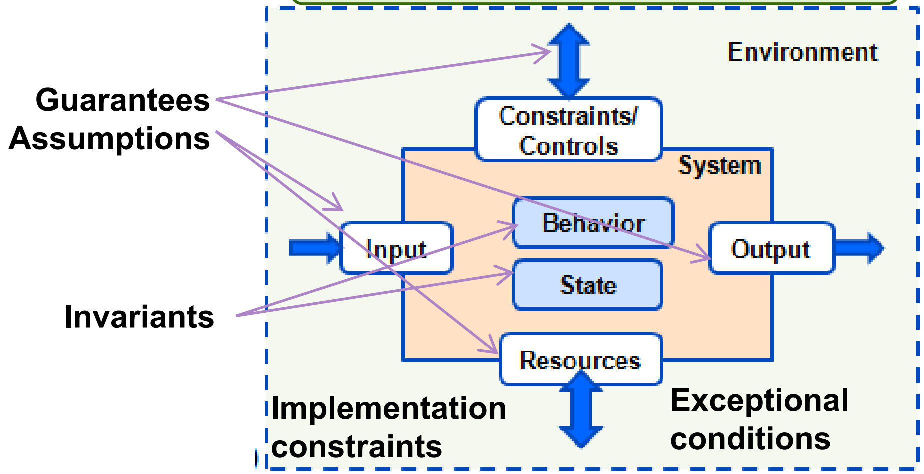
3. When piston stop is received, the **system** shall stop piston movement within 0.01 seconds.
4. The **system** shall always stop the piston at the bottom or top of the chamber.

Importance of understanding system boundary
Multiple layers of system specification

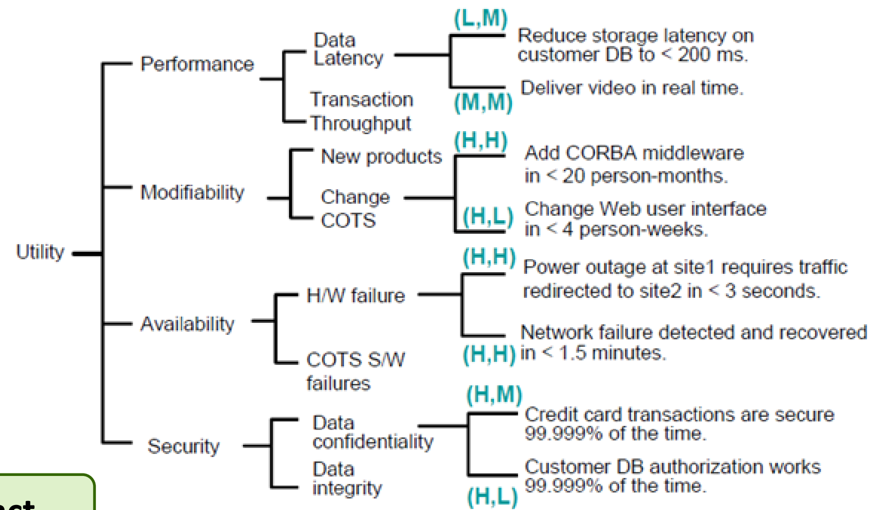
Typical requirement documents span multiple levels of a system architecture
We have effectively specified a partial architecture

Three Dimensions of Requirement Coverage

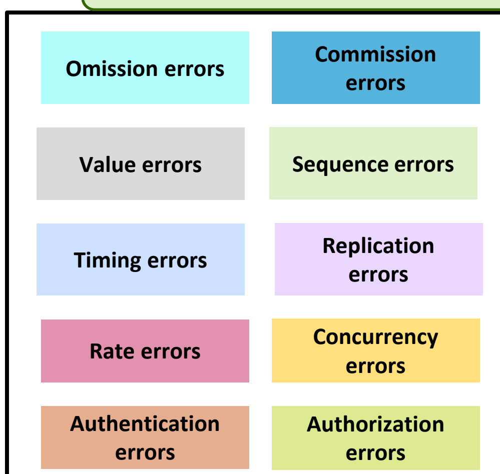
System interactions, state, behavior



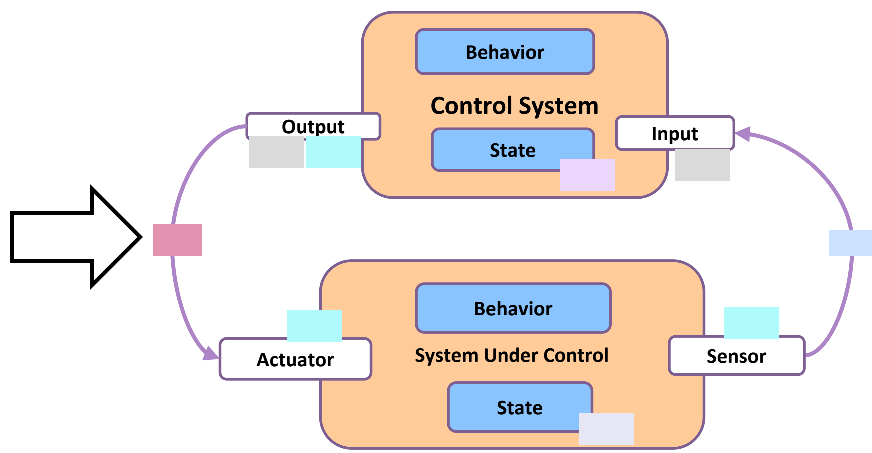
Design & operational quality attributes



Fault contributors & impact



Fault Propagation Taxonomy



Outline

Challenges in Safety-critical Software-intensive Systems

Four Pillar Improvement Strategy

Verifiable System Requirement Specification

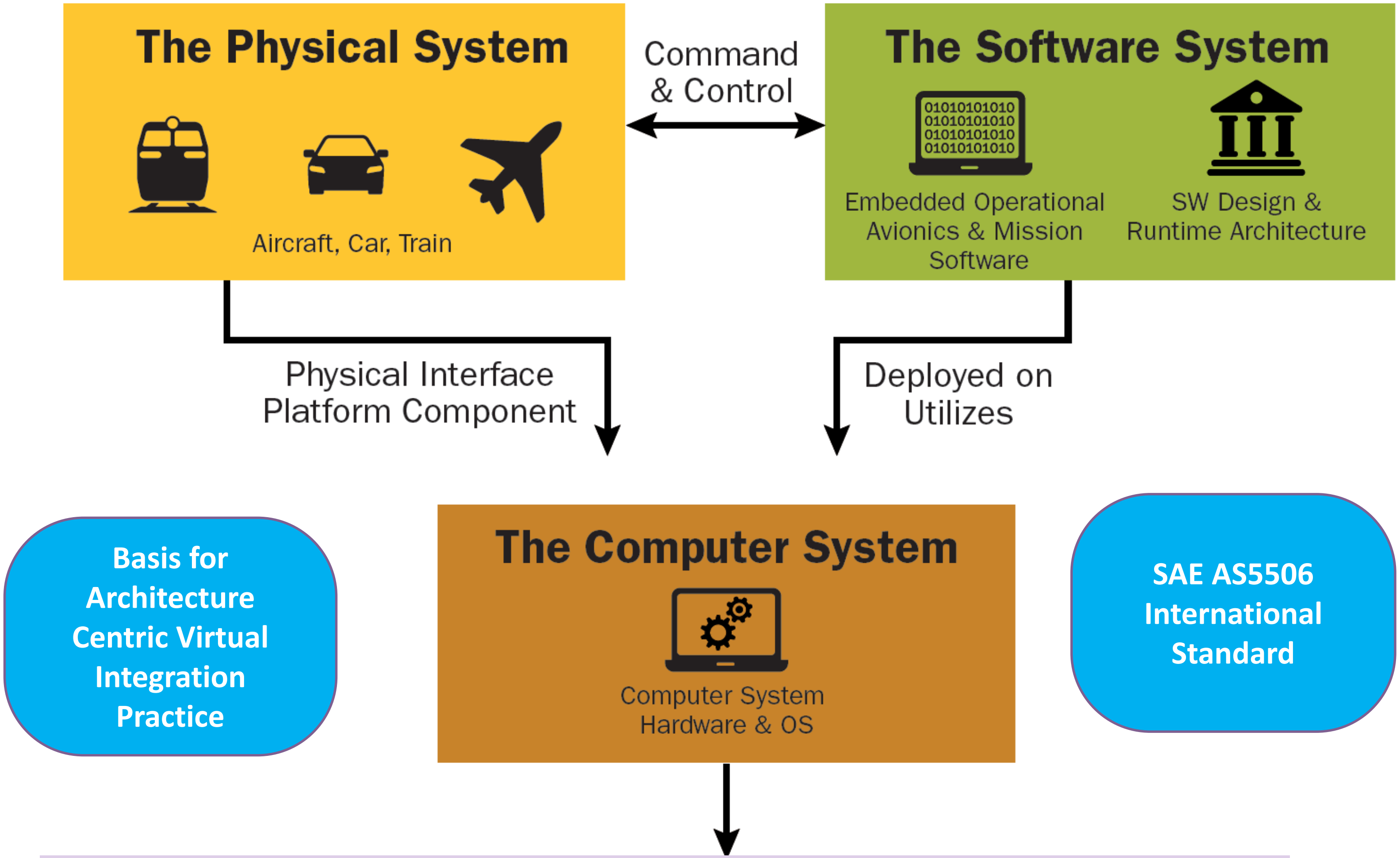
▶ Architecture-Centric Virtual System Integration

Compositional Verification to Complement Testing

Incremental System Assurance throughout Development

Conclusion

Architecture Analysis & Design Language (AADL) Enables Industry-Wide Virtual Integration and Assurance Approach



AADL captures software-reliant, mission and safety critical system architectures in a computable model to discover system level problems early

SAE AADL Standard Suite (AS-5506 series)

Core AADL language standard (V2.2-Jan 2017, V1-Nov 2004)

- Strongly typed language with well-defined execution and communication semantics
- Textual and graphical notation
- Revision V3 in progress: Interface composition, system configuration, binding, type system unification

Standardized AADL Annex Extensions
Error Model language for safety, reliability, security analysis
ARINC653 extension for partitioned architectures
Behavior Specification Language for modes and interaction behavior
Data Modeling extension for interfacing with data models (UML, ASN.1,)
AADL Runtime System & Code Generation

AADL Annexes in Progress
Network Specification Annex
Cyber Security Annex
Requirements Definition and Assurance Annex
Synchronous System Specification Annex
Hybrid System Specification Annex
System Constraint Specification Annex

System Level Fault Root Causes

Addressed by AADL Concepts with Well Defined Semantics

Violation of Data Stream Assumptions

- Stream miss rates, Mismatched data representation
- Latency jitter & age

End-to-end flows for latency analysis
Sampling & queued ports
Mid-frame & frame-delayed connections
Port connection consistency

Partitions as Isolation Regions

- Space, time, and bandwidth partitioning
- Isolation not guaranteed due to undocumented resource sharing

Process and virtual processor to model partitioned architectures

Virtualization of Time & Resources

- Logical vs. physical redundancy
- Time stamping of data & asynchronous systems

Virtual processors & virtual buses
Multiple time domains

Inconsistent System States & Interactions

- Communication of states and events
- Concurrency & redundancy management

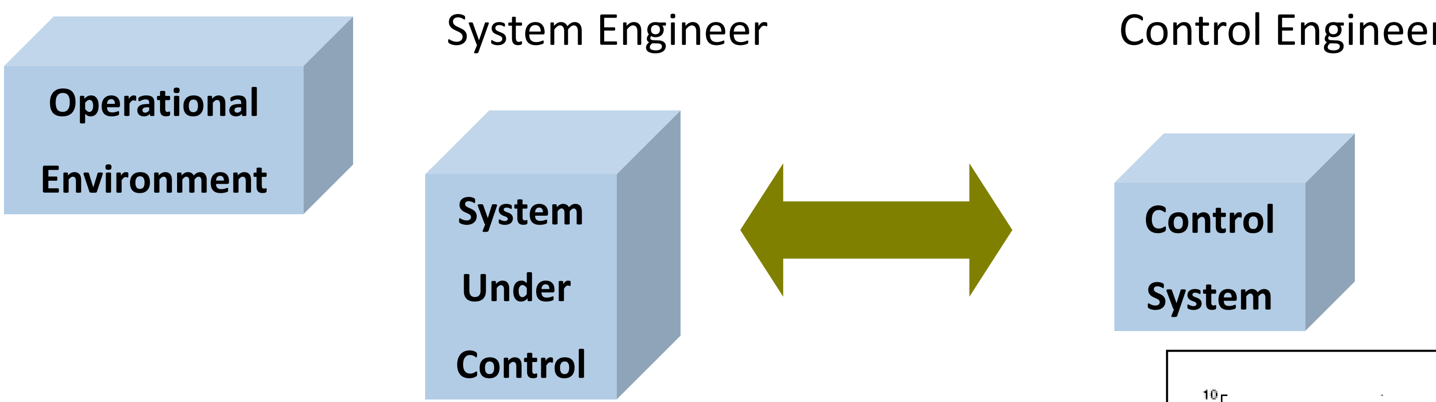
Operational and failure modes
Interaction behavior specification
Dynamic reconfiguration
Fault detection, isolation, recovery

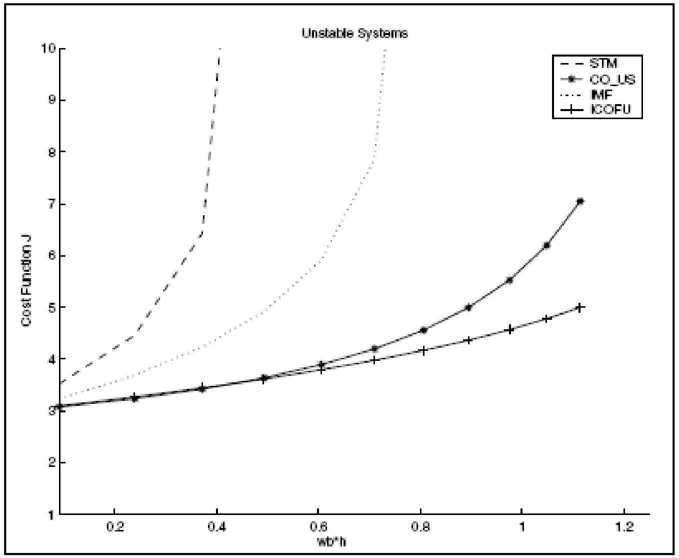
Shared Resource Management

- Processor, memory & network resources
- Unmanaged computer system resources

Resource allocation & deployment configurations
Resource budget analysis & scheduling analysis

Multi-Fidelity End-to-end Latency in Control Systems





Common latency data from system engineering

- Processing latency
- Sampling latency
- Physical signal latency

Impact of Scheduler Choice on Controller Stability
A. Cervin, Lund U., CCACSD 2006

Software-Based Latency Contributors

Execution time variation: algorithm, use of cache

Processor speed

Resource contention

Preemption

Shared variable communication

Rate group optimization

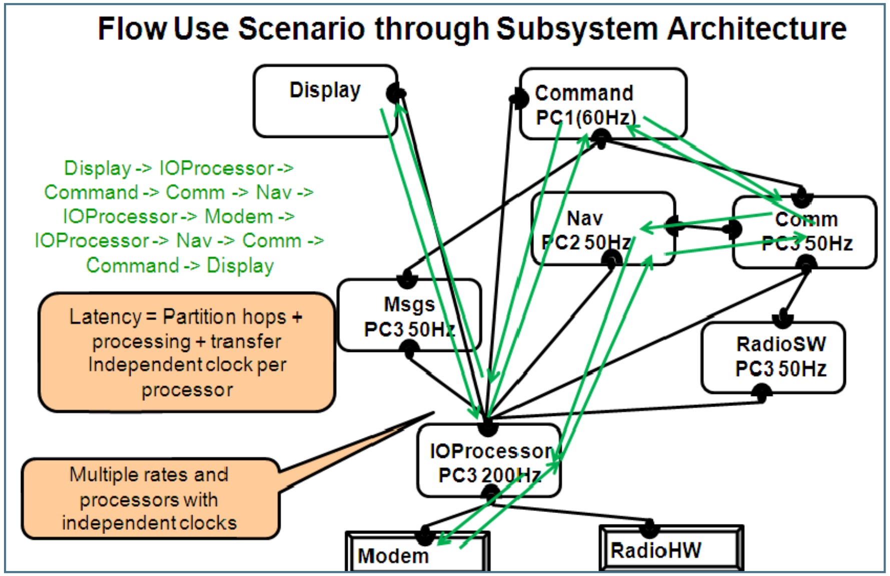
Protocol communication delay

Partitioned architecture

Migration of functionality

Fault tolerance strategy

Synchronized time domains



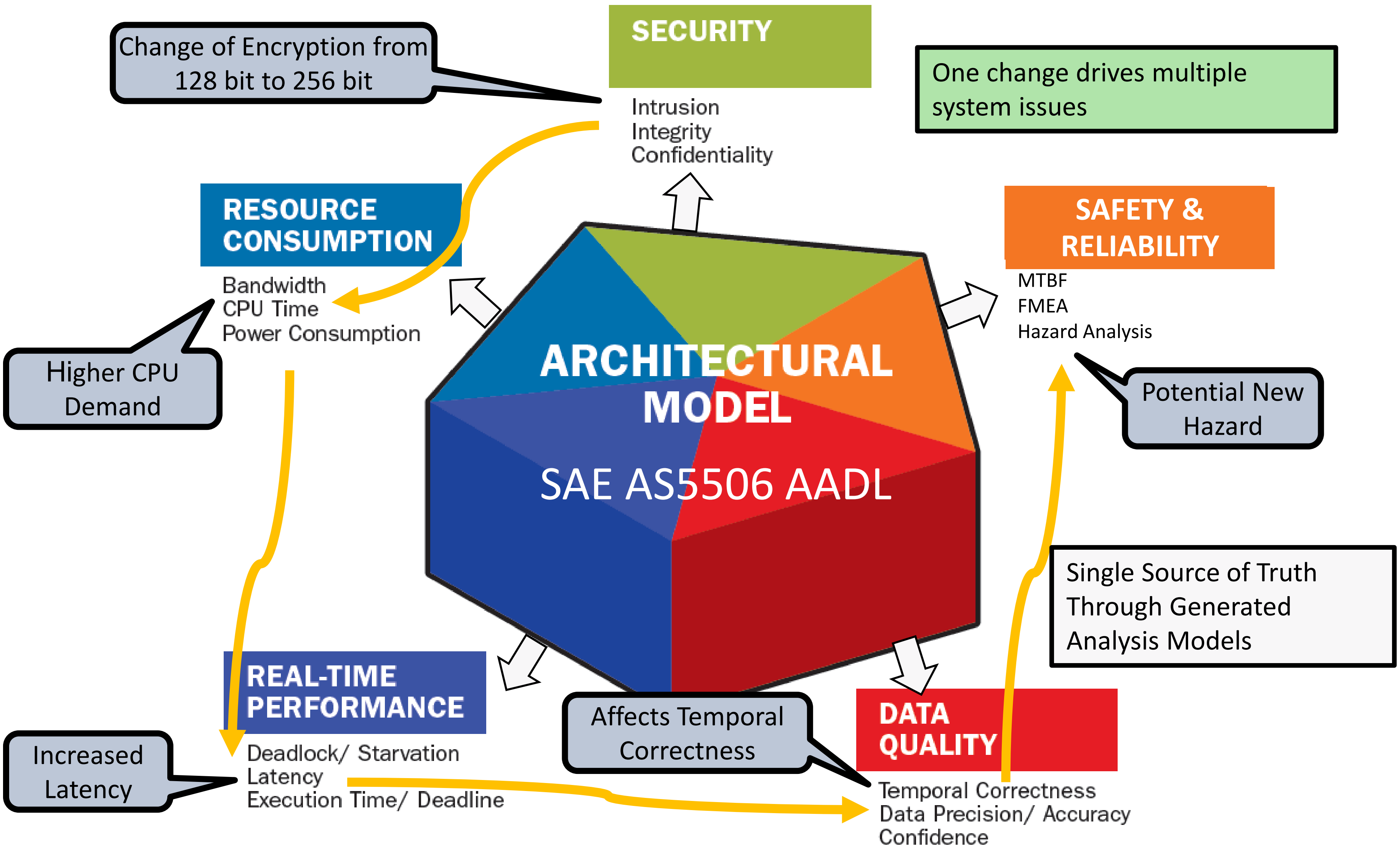
From Customer Design Document

"The 200 Hz update rate was used because the MUX data needed to be processed at twice the rate of the fastest channel to avoid a race condition. Because channel 3 operates at 100 Hz, the IO processor had to operate at 200 Hz. ***The race condition has been fixed by double-buffering data,*** *but the IO processor execution rate was left at 200 Hz to reduce latency of MUX data."*

Outline

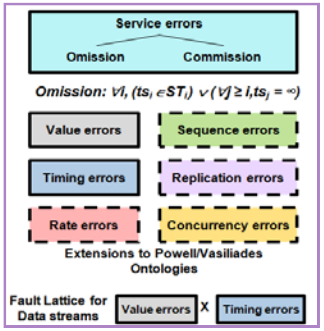
- Challenges in Safety-critical Software-intensive Systems
- Four Pillar Improvement Strategy
- Verifiable System Requirement Specification
- Architecture-Centric Virtual System Integration
- ▶ Compositional Verification to Complement Testing
- Incremental System Assurance throughout Development
- Conclusion

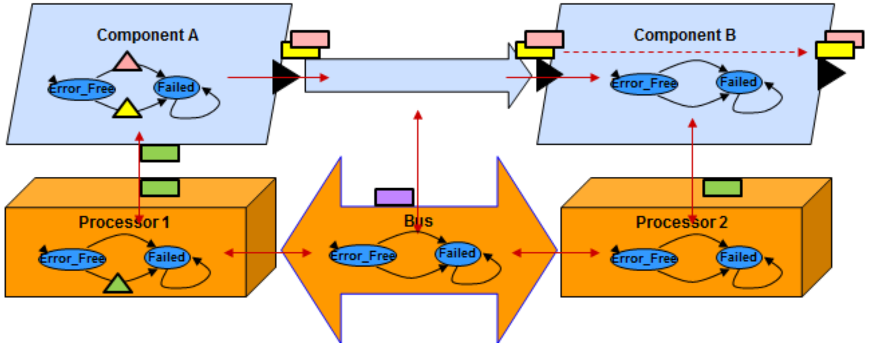
Change Impact Across Multiple Quality Dimensions



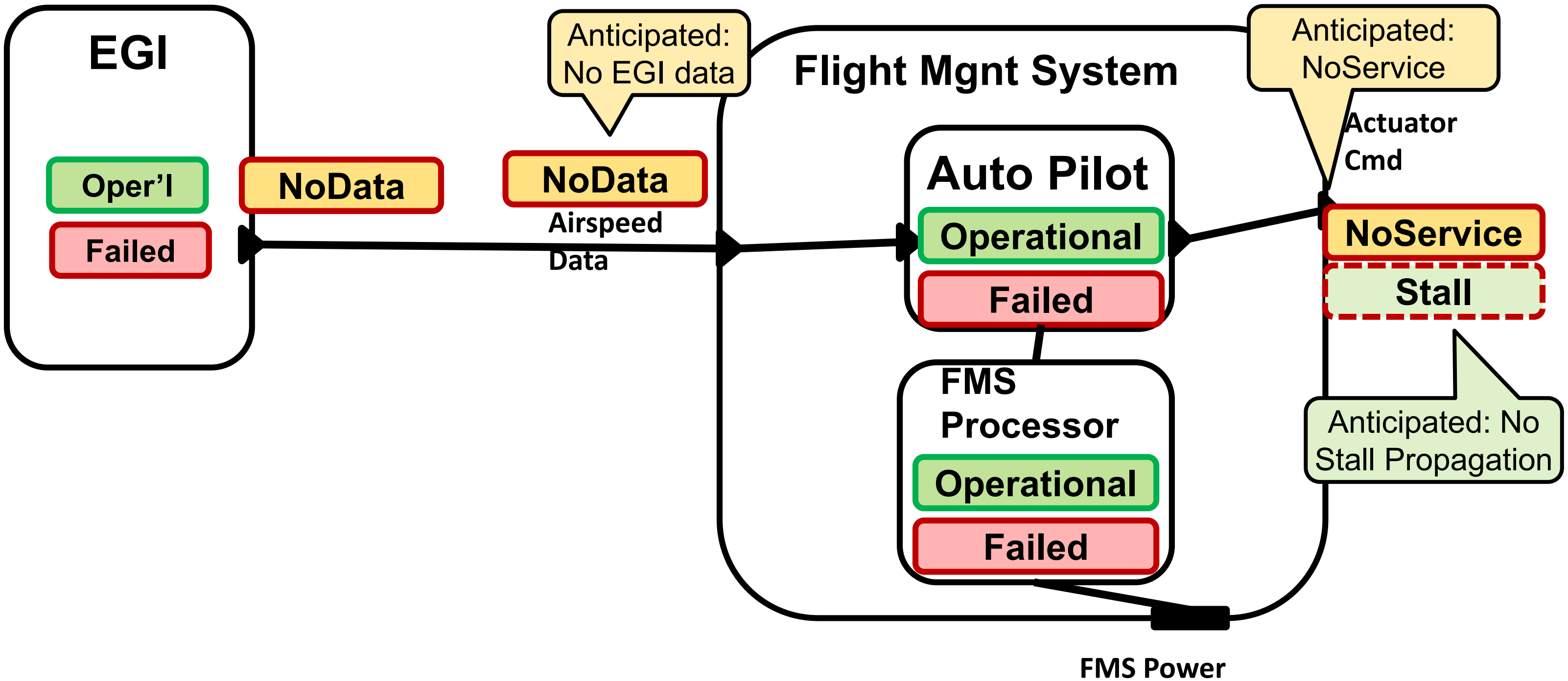
Error Model V2 Annex

- Focus on fault interaction with other components
  - Probabilistic error sources, sinks, paths and transformations
  - Fault propagation and Transformation Calculus (FPTC) from York U.
- Focus on fault behavior of components
  - Probabilistic typed error events, error states, propagations
  - Voting logic, error detection, recovery, repair
- Focus on fault behavior in terms of subcomponent fault behaviors
  - Composite error behavior state logic maps states of parts into (abstracted) states of composite
- Typed token system
  - Fault effect taxonomy
  - Domain specific fault types



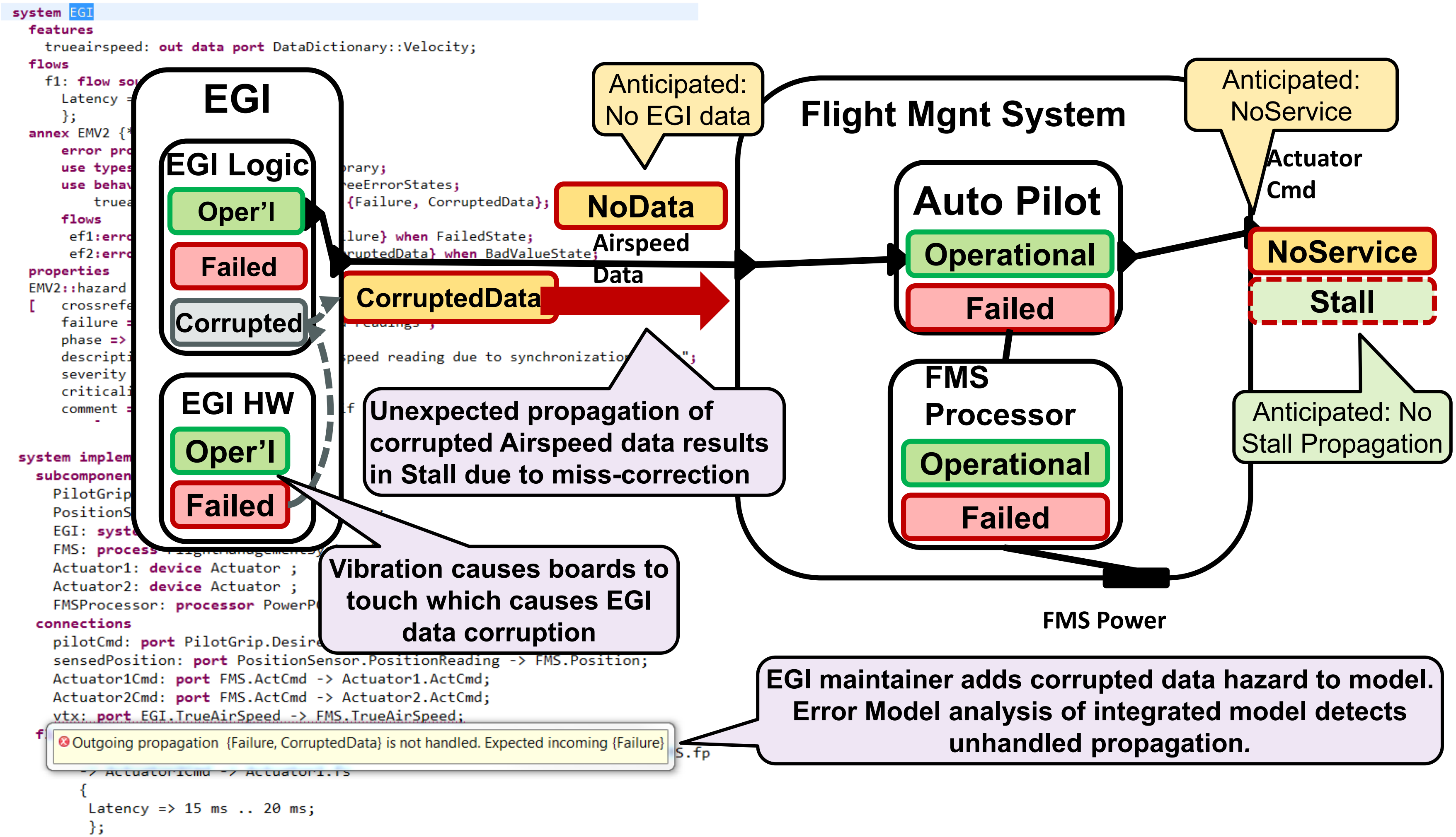


Original Preliminary System Safety Analysis (PSSA)



System engineering activity with focus on failing components

Discovery of Unexpected PSSA Hazard through Repeated Virtual Integration



Automated FMEA Experience

Failure Modes and Effects Analyses are rigorous and comprehensive reliability and safety design evaluations

- Required by industry standards and Government policies
- When performed manually are usually done once due to cost and schedule
- When automated allows for
  - multiple iterations from conceptual to detailed design
  - Tradeoff studies and evaluation of alternatives
  - Early identification of potential problems

ID	Item	Initial State	Initial Failure Mode	1st Level Effect	Transition	2nd Level Effect	Transition	3rd Level Effect	Severity	M
1	Sat_Bus	Working	Failure	Failed		Failed	Recovery	Working		Workir
1	Sat_Payload	Working		Working	Bus failure causes payload transition	Standby		Standby	Bus Recovery Causes Payload Transition	Workir
2	Sat_Bus	Working		Working		Working	5			
2	Sat_Payload	Working	Failure	Failed	Recovery	Working	5			

Largest analysis of satellite to date consists of 26,000 failure modes

- Includes detailed model of satellite bus
- 20 states perform failure mode
- Lonest failure mode sequences have 25 transitions (i.e., 25 effects)

Myron Hecht, Aerospace Corp.
Safety Analysis for JPL, member of DO-178C committee

Time Sensitive Engine Control Problem

Stepper motor (SM) controls a valve

- Commanded to achieve a specified valve position
  - Fixed position range mapped into units of SM steps
- New target positions can arrive at any time
  - SM immediately responds to the new desired position

Safety hazard due to software design

- Execution time variation results in missed steps
- Leads to misaligned stepper motor position and control system states
- Sensor feedback not granular enough to detect individual step misses

Software modeled and verified in SCADE
Full reliance on SCADE of SM & all functionality
Problems with missing steps not detected

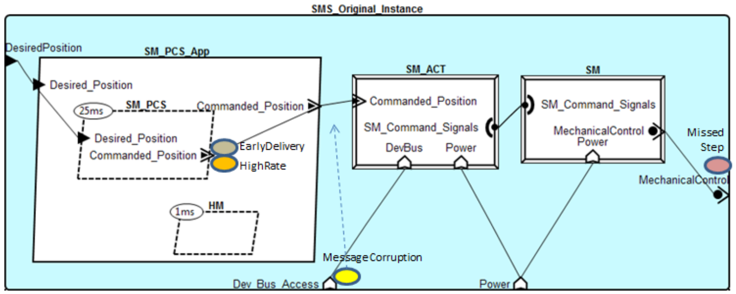
Software tests did not discover the issue
Time sensitive systems are hard to test for.

Two Customer Proposed Solutions
Sending of data at 12ms offset from dispatch
Buffering of command by SM interface
No analytical confidence that the problem will be addressed

Analysis Results and Solution

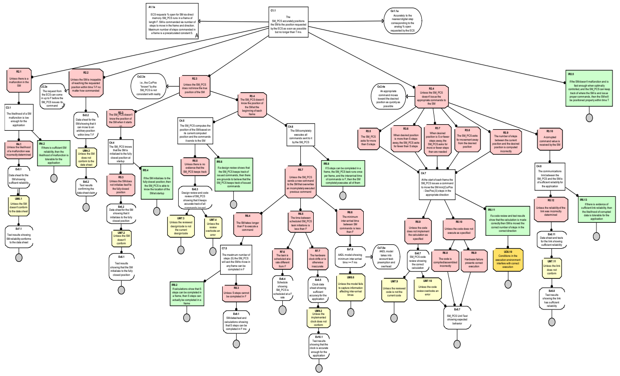
Architecture Fault Model Analysis

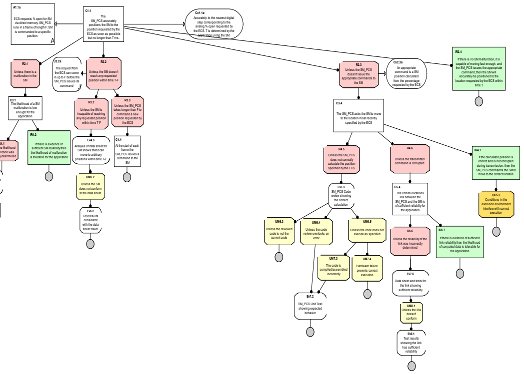
- Fault impact analysis identifies multiple sources of missed steps
  - Early arrival of step increment commands
  - Step increment command rate mismatch
  - Transient message corruption or loss
- Understanding of error cause
  - When is early too early
  - Guaranteed delivery assumption for step increment commands



MissedStep	Original Design	Fixed Send Time	Buffered Command	Position Command
SMS logical failures	EarlyDelivery HighRate	HighRate	HighRate	
SMS mechan-ical failures	ActuatorFailure StepperMotorFailure	ActuatorFailure StepperMotorFailure	ActuatorFailure StepperMotorFailure	ActuatorFailure StepperMotorFailure
Transient comm failures	MessageCorruption MessageLoss	MessageCorruption MessageLoss	MessageCorruption MessageLoss	
Mechanical failures in Op Environment	ECUFailure PowerLoss ValveFailure	ECUFailure PowerLoss ValveFailure	ECUFailure PowerLoss ValveFailure	ECUFailure PowerLoss ValveFailure

Assurance Case Confidence Maps





Integrated Safety and Security Engineering

Safety perspective of safety-critical systems

- Integrated modular avionics: ARINC653 partitions
  - Space and time partitioning of shared resource
- Safety levels and information/control flow
  - Functional analysis: levels of coverage (MCDC for Level A)
- Fault detection, isolation, recovery (FDIR)
- Zero defect assumption not valid for software
  - New focus: analytic redundancy, resilience

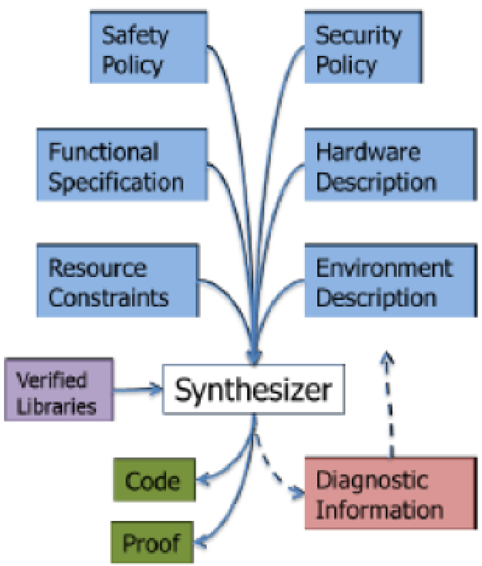
Cyber security issues

- Malicious external interactions with system
  - Via established interfaces, denial of service
- Unauthorized replacement of system component
  - Need for continuous authentication and isolation within system

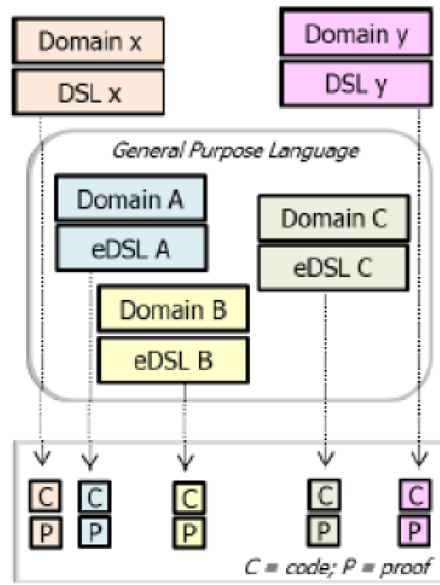
Synthesize & Verify High-Assurance Systems

High Assurance Cyber Military Systems **Dr. Lindermann April 2018 Keynote**

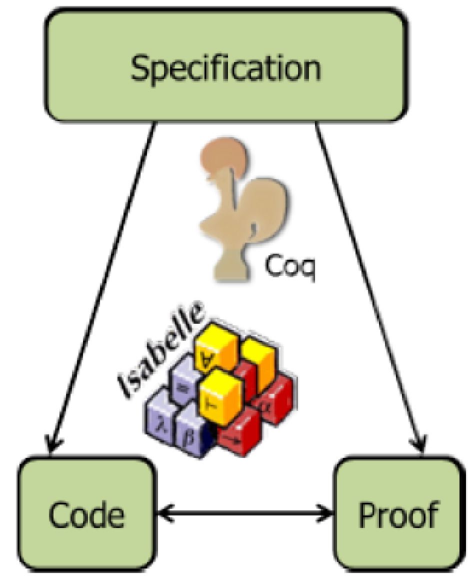
Code Synthesis



Domain Specific Languages (DSLs)



Interactive Theorem Prover as PL



- **Research Challenges**
  - Synthesis of attack-resilient control systems
  - Synthesis of operating systems code
  - Specification languages: function, environment,
  - hardware, resources
  - Composition/Proof engineering
  - Scaling
  - Attack/fault response
  - V&V of complete system

Outline

Challenges in Safety-critical Software-intensive Systems

Four Pillar Improvement Strategy

Verifiable System Requirement Specification

Architecture-Centric Virtual System Integration

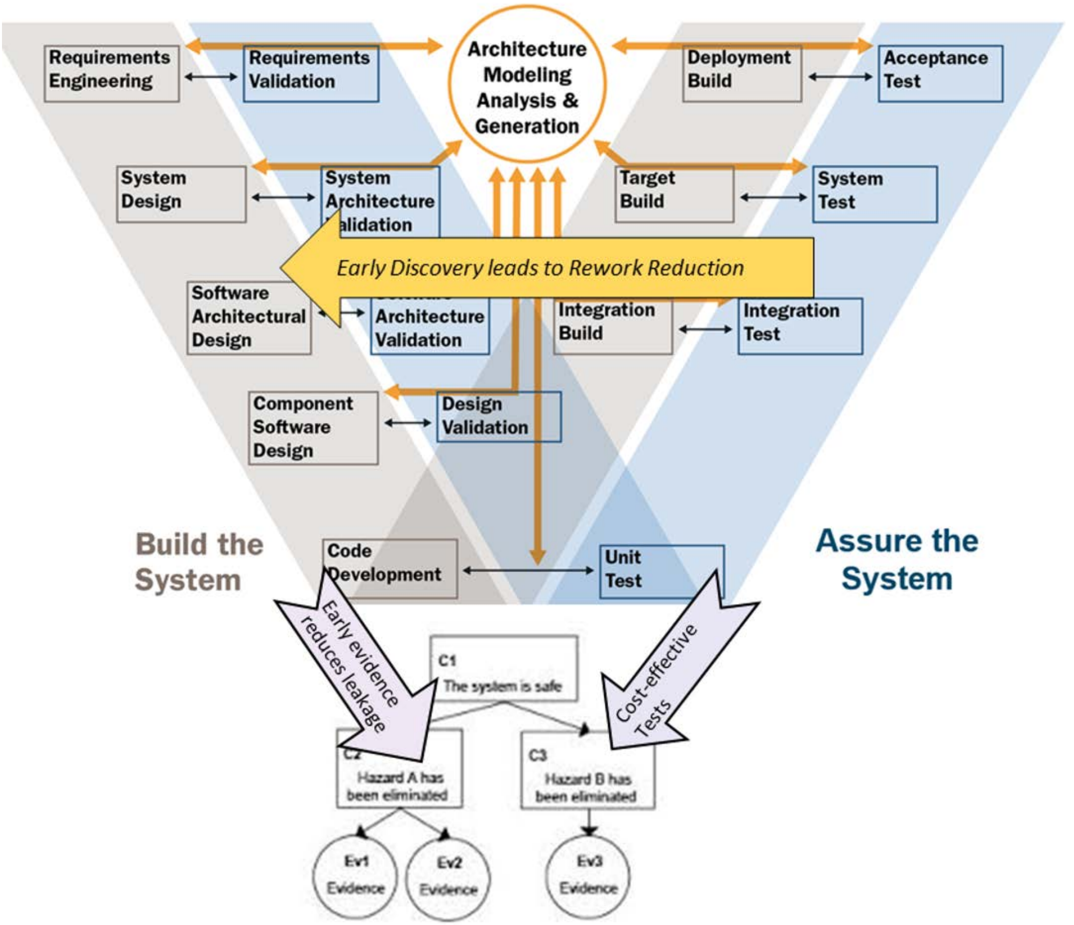
Compositional Verification to Complement Testing

Incremental System Assurance throughout Development

Conclusion

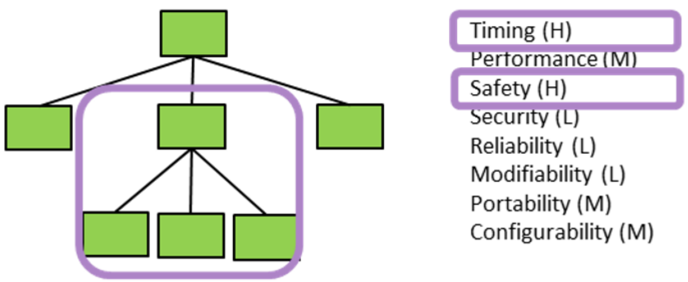
Three Dimensions of Incremental Assurance

1. Incremental assurance through virtual system integration for early discovery

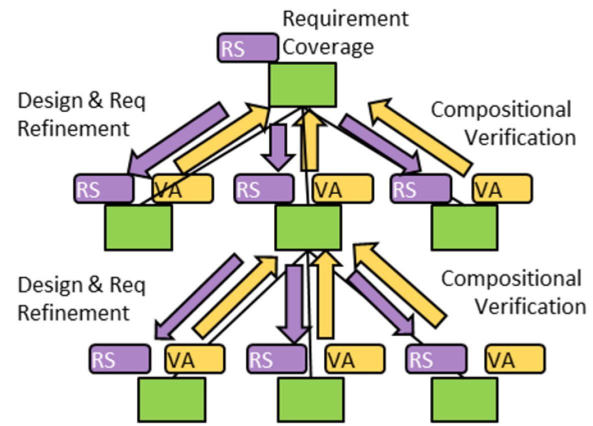


Auto-generated Assurance Cases

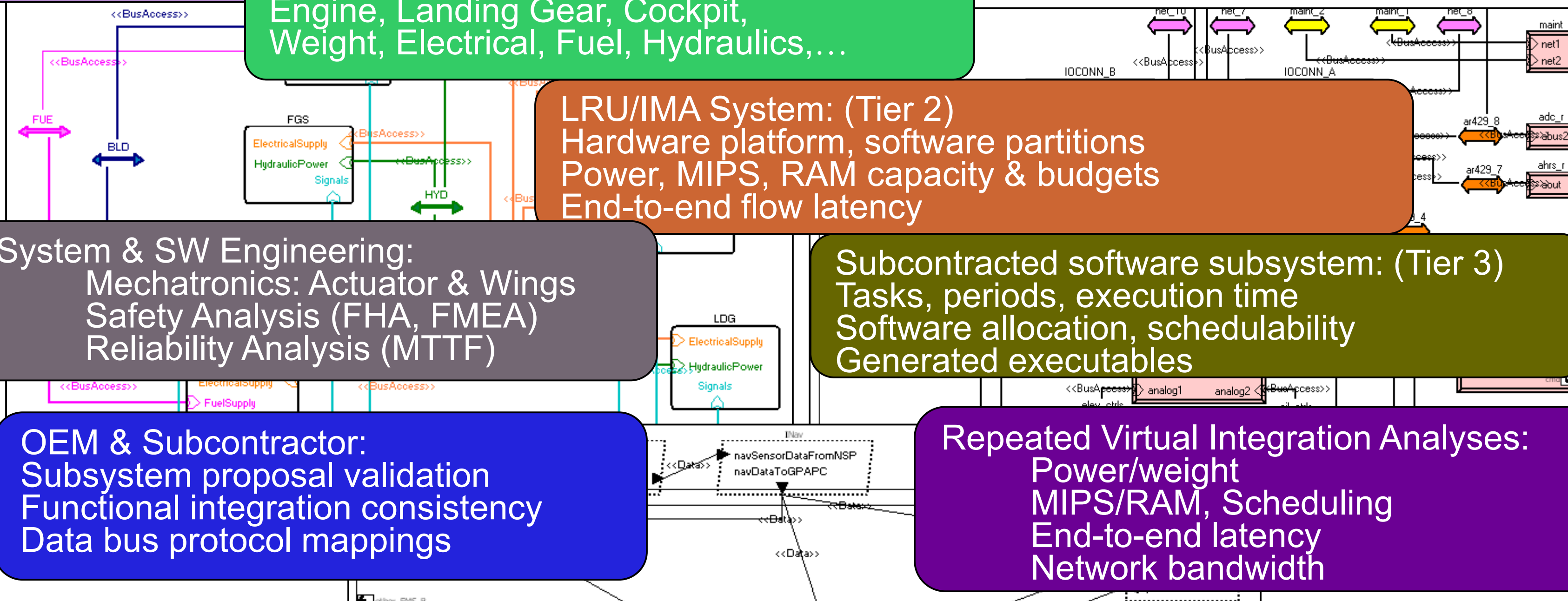
2. Priority focused architecture design exploration for high payoff



3. Contract-based Compositional Verification



Early Discovery and Incremental V&V through Virtual System Integration (SAVI Proof of Concept in 2009)



Aircraft: (Tier 0)

Aircraft system: (Tier 1)
Engine, Landing Gear, Cockpit,
Weight, Electrical, Fuel, Hydraulics,…

LRU/IMA System: (Tier 2)
Hardware platform, software partitions
Power, MIPS, RAM capacity & budgets
End-to-end flow latency

System & SW Engineering:
Mechatronics: Actuator & Wings
Safety Analysis (FHA, FMEA)
Reliability Analysis (MTTF)

Subcontracted software subsystem: (Tier 3)
Tasks, periods, execution time
Software allocation, schedulability
Generated executables

OEM & Subcontractor:
Subsystem proposal validation
Functional integration consistency
Data bus protocol mappings

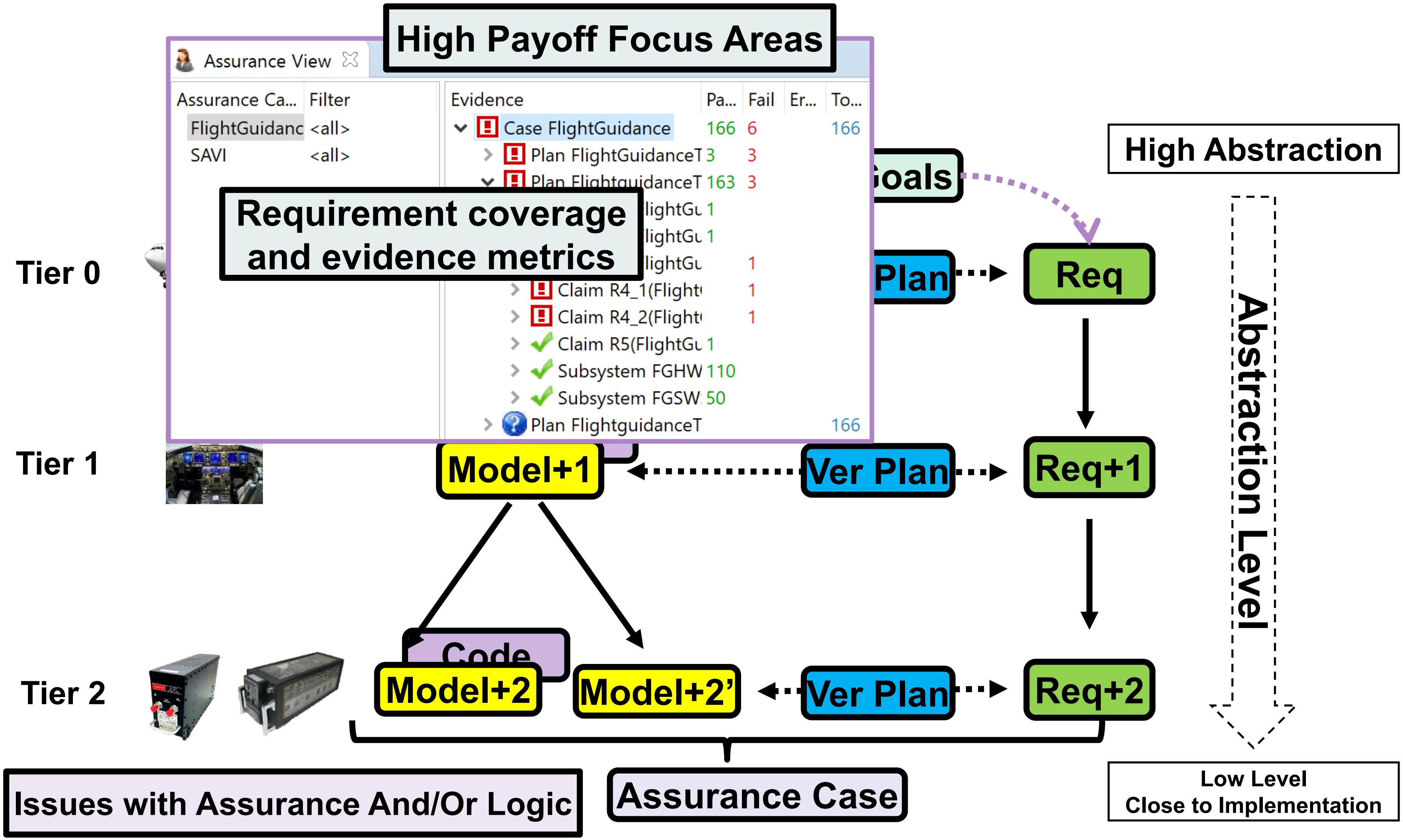
Repeated Virtual Integration Analyses:
Power/weight
MIPS/RAM, Scheduling
End-to-end latency
Network bandwidth

Proof of Concept Demonstration and Transition by Aerospace industry initiative

- Architecture-centric model-based software and system engineering
- Multi notation, multi team model repository & standardized model interchange
- Single source of truth challenge

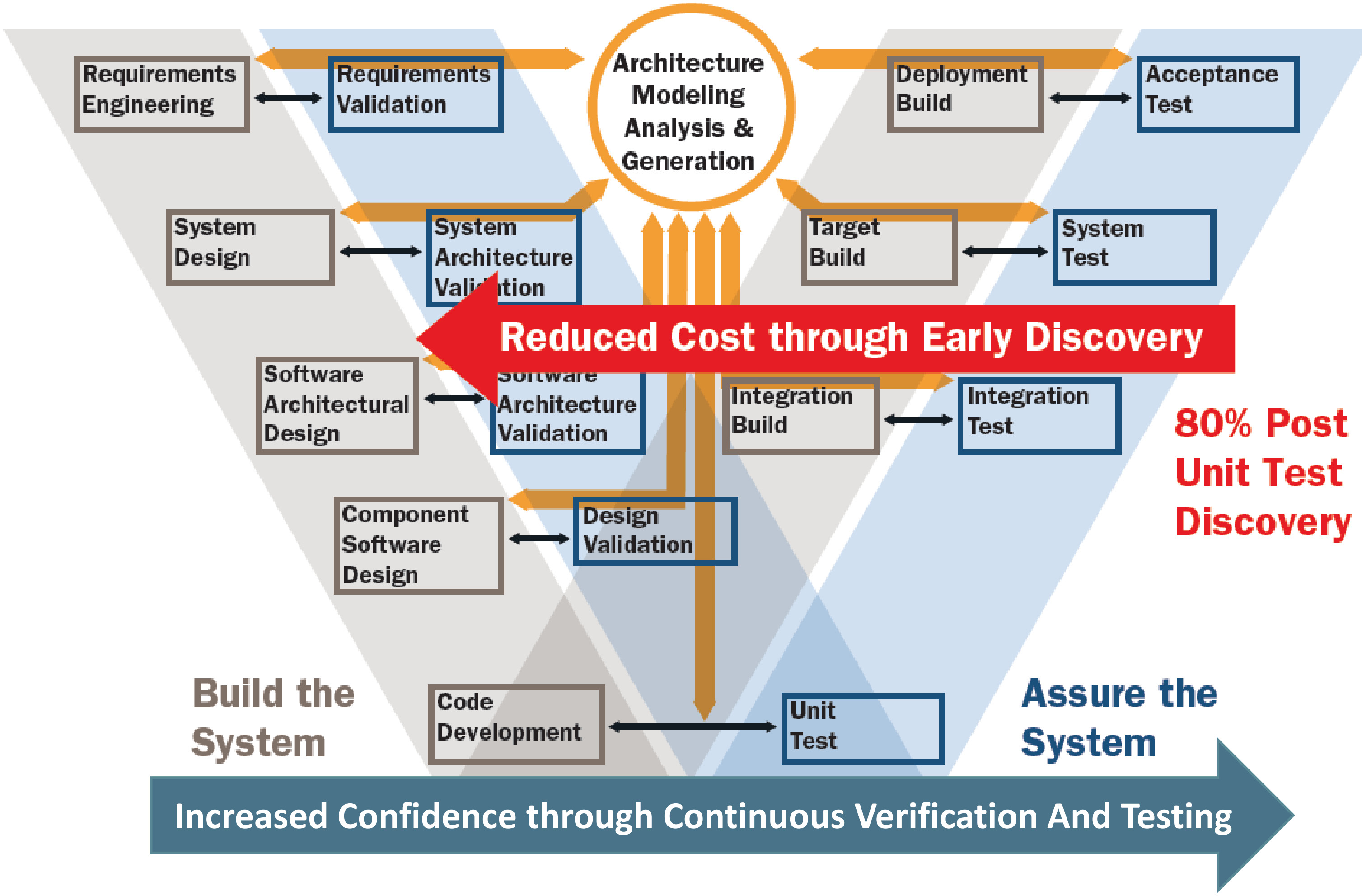
- Multi-tier system & software architecture (in AADL)
- Incremental end-to-end validation of system properties

Automated Incremental Assurance Workbench



Outline

Benefits of Virtual System Integration & Incremental Lifecycle Assurance



References

Four Pillar Qualification Improvement Framework

Blog post https://insights.sei.cmu.edu/sei_blog/2013/06/improving-safety-critical-systems-with-a-reliability-validation-improvement-framework.html

Four pillars white paper https://resources.sei.cmu.edu/library/asset-view.cfm?assetid=47791

Full TR https://resources.sei.cmu.edu/library/asset-view.cfm?assetID=34069

SAVI ROI https://resources.sei.cmu.edu/library/asset-view.cfm?assetid=517157

Stepper Motor Case Study

Paper https://hal.archives-ouvertes.fr/hal-01292322/document

TR https://resources.sei.cmu.edu/library/asset-view.cfm?assetid=435051

Incremental Assurance (ALISA) in OSATE

online doc https://rawgit.com/osate/alisa/develop/org.osate.alisa.help/contents/00-Main.html

Paper: https://hal.archives-ouvertes.fr/hal-01289468

ALISA Example models https://github.com/osate/alisa-examples

Latency Analysis

Paper https://e-archivo.uc3m.es/bitstream/handle/10016/19688/incremental_REACTION_2014.pdf

Online doc https://github.com/osate/osate2/blob/develop/analyses/org.osate.analysis.flows/help/markdown/latency.md

Other papers

ReqSpec Notation SEI TR: http://resources.sei.cmu.edu/library/asset-view.cfm?assetid=464370

Two JMR case study reports: https://resources.sei.cmu.edu/library/asset-view.cfm?assetid=447176

http://resources.sei.cmu.edu/asset_files/specialreport/2015_003_001_447187.pdf

JMR paper https://www.army.mil/e2/c/downloads/414601.pdf

HILT 2016 http://sigada.org/conf/hilt2016/paper-McGregor.pdf

ERTS 2016 https://hal.archives-ouvertes.fr/hal-01292290/

Contact Information

Peter Feiler
SEI Fellow
Telephone: +1 412.268.7790
Email: phf@sei.cmu.edu

U.S. Mail:
Software Engineering Institute
Carnegie Mellon University
4500 Fifth Avenue
Pittsburgh, PA 15213-3890