# Incremental Lifecycle Assurance of Critical Systems

Peter Feiler

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA  15213

# Incremental Lifecycle Assurance Objectives

Improve critical system assurance through

- Improved requirement quality through coverage and managed uncertainty
- Improved evidence quality through compositional analytical verification
- Measurably reduced certification related rework cost through virtual integration and verification automation

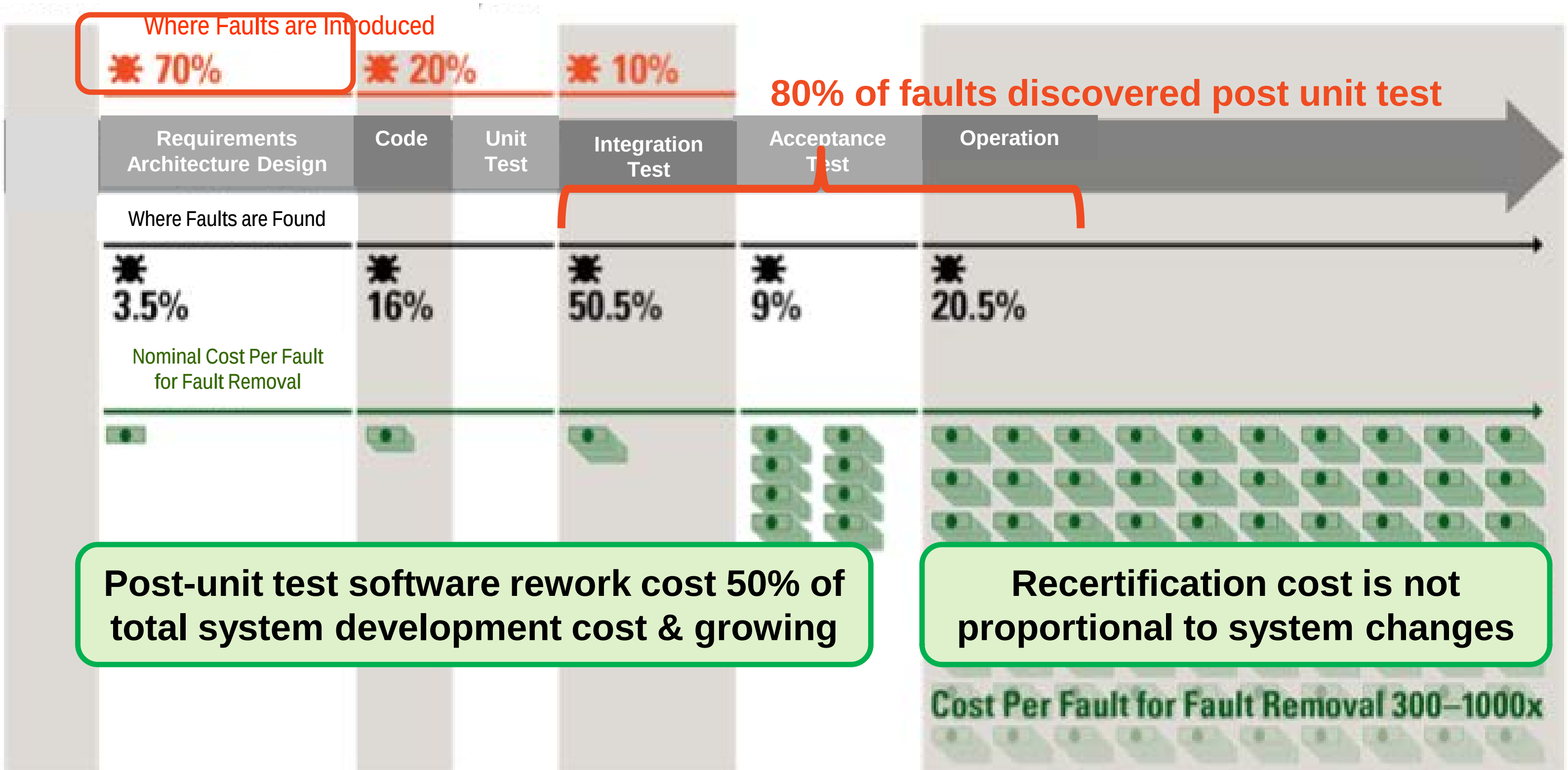
# Outline

**Critical system assurance challenges**

Incremental life cycle assurance approach

Year One Accomplishments

# Critical System Assurance Challenges




**Post-unit test software rework cost 50% of total system development cost & growing**

**Recertification cost is not proportional to system changes**

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

**Years between labor-intensive system safety assessments**
**Software as major hazard source often ignored**

# Value of Requirement Uncertainty Awareness

## Textual requirement quality statistics

- Current requirement engineering practice relies on stakeholders traceability and document reviews resulting in high rate of requirement change.

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

**NIST Study**

## Managed awareness of requirement uncertainty reduces requirement changes by 50%

- 80% of requirement changes from development team
- Expert requirement uncertainty assessment
  - Volatility, Impact, Precedence, Time criticality
- Focus on high 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 |

Figure 8. Precedence measurement scale

**Rolls Royce Study**

# Mixture of Requirements & Architecture Design Constraints

## Textual 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.

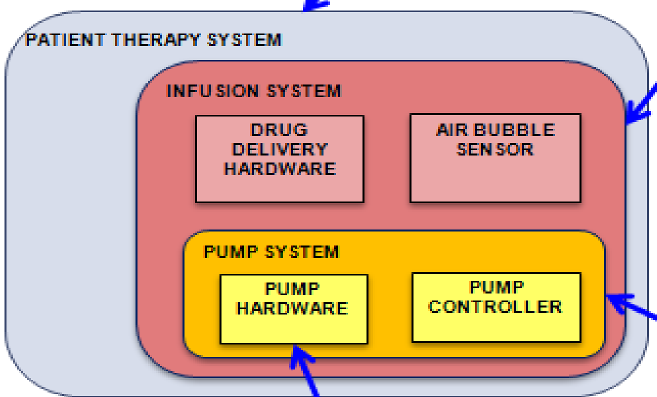
When piston stop is received, the **system** shall stop piston movement within 0.01 seconds.

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

## Same Requirements Mapped to an Architecture Model

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.




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

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

**Importance of understanding system boundary**

**We have effectively specified a system partial architecture**

U Minnesota Study

# Outline

Assurance challenges

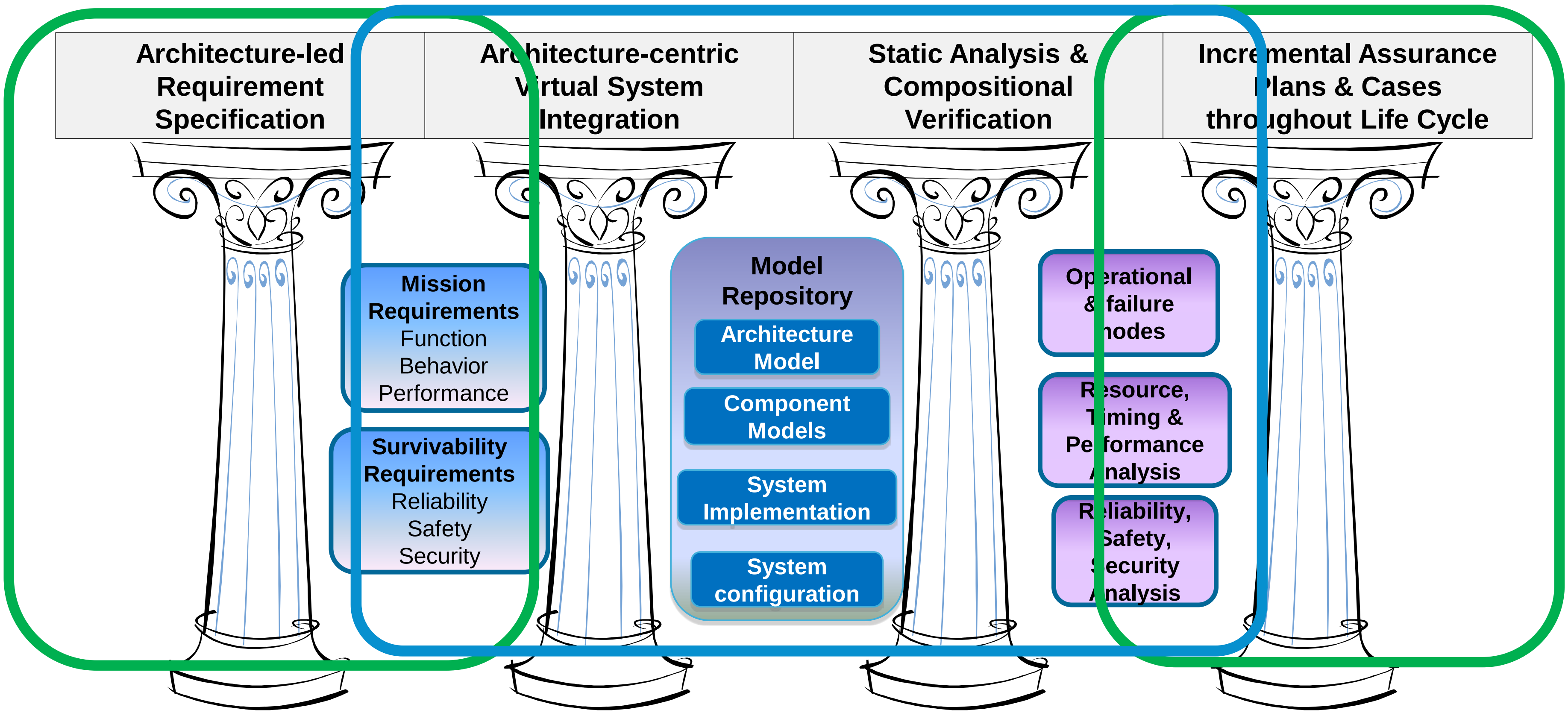
**Incremental life cycle assurance approach**

Year One Accomplishments

# Assurance & Qualification Improvement Strategy

**Assurance: <u>Sufficient</u> <u>evidence</u> that a <u>system implementation</u> meets <u>system requirements</u>**

*2010 SEI Study for AMRDEC Aviation Engineering Directorate*

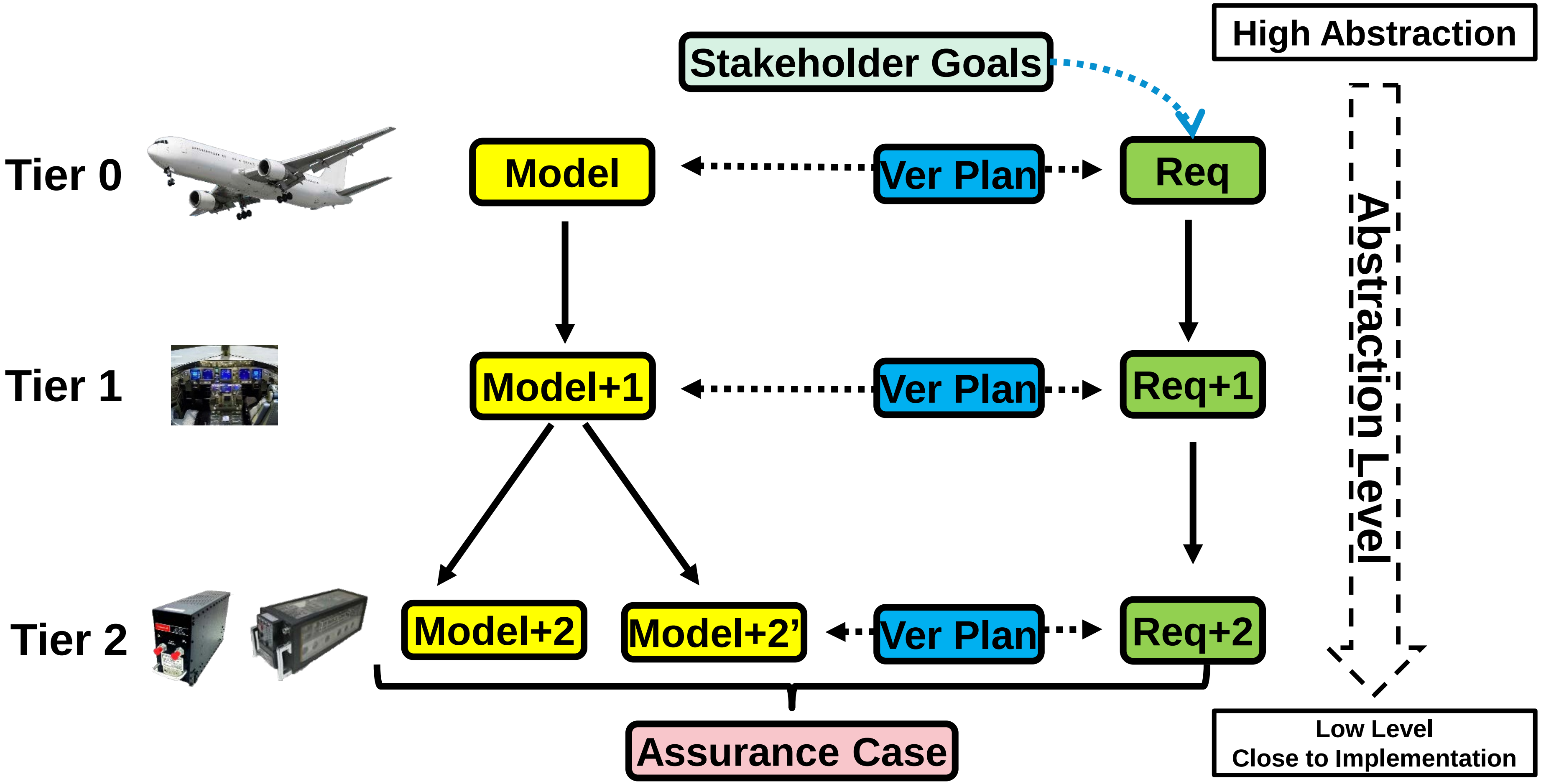



**Early Problem Discovery through Virtual System Integration & Analysis**

**Improved Assurance through Better Requirements & Automated Verification**

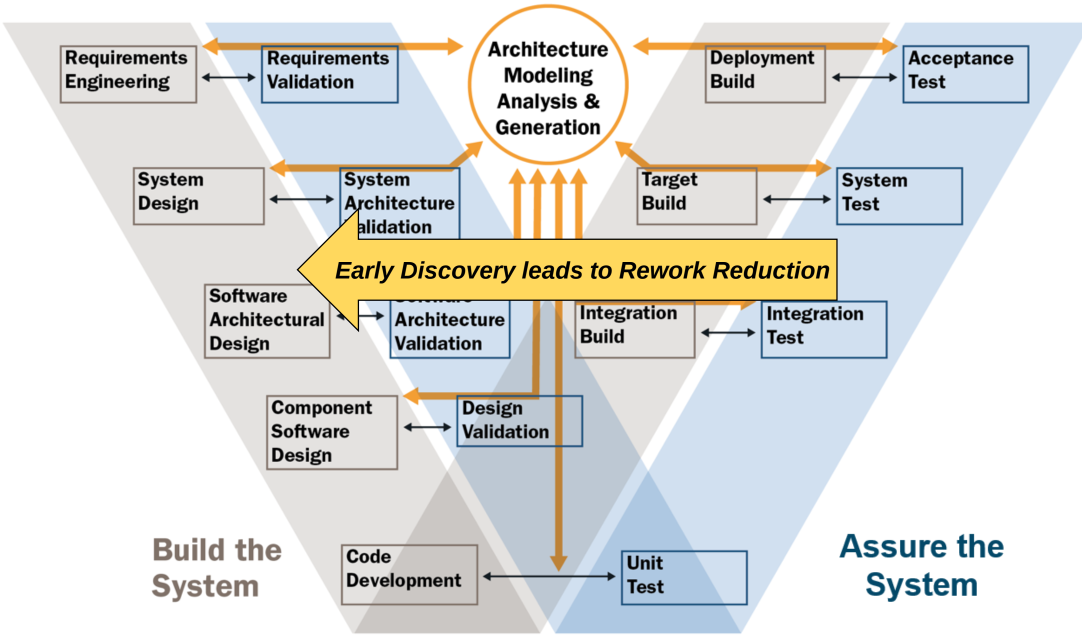
# Automated Incremental Assurance Workbench

**Identify Assurance Hotspots Throughout Lifecycle**



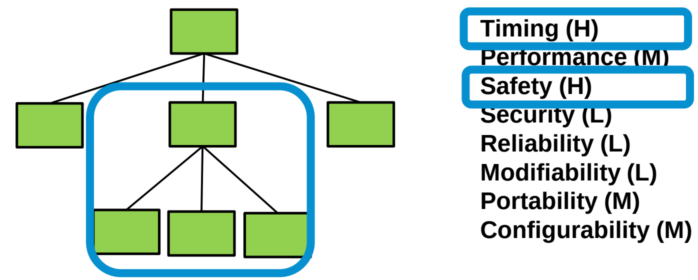
# Three Dimensions of Incremental Assurance

**Incremental assurance through virtual system integration for early discovery**
Return on Investment study by SAVI*

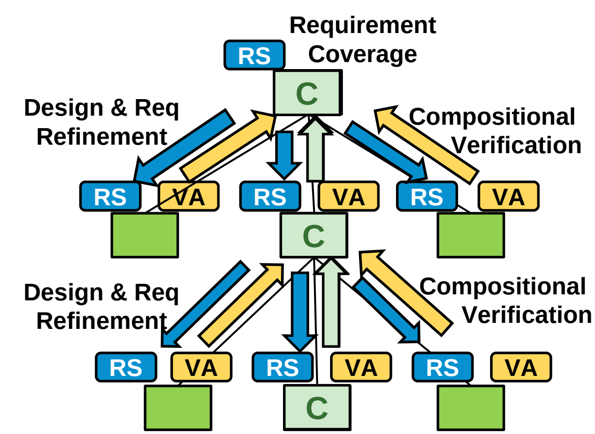



*System Architecture Virtual Integration (SAVI) Aerospace industry initiative

**Priority focused architecture design exploration for high payoff**
Measurable improvement (Rolls Royce)

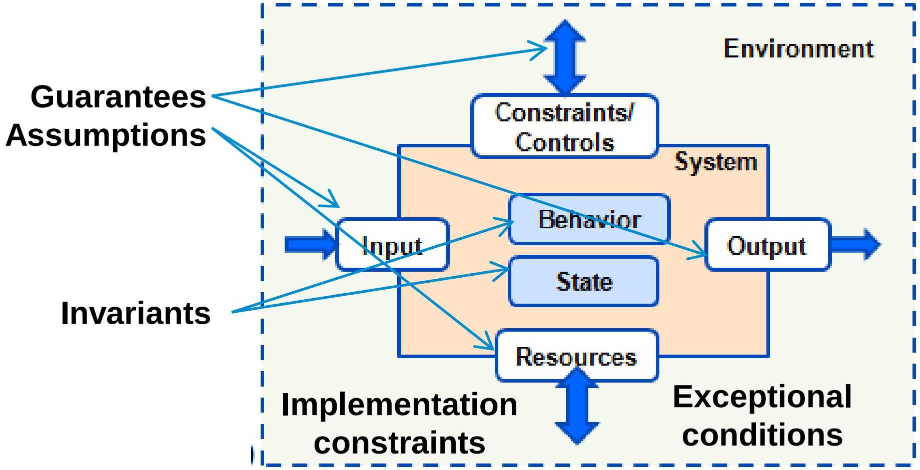



**Compositional verification and partitions to limit assurance impact**

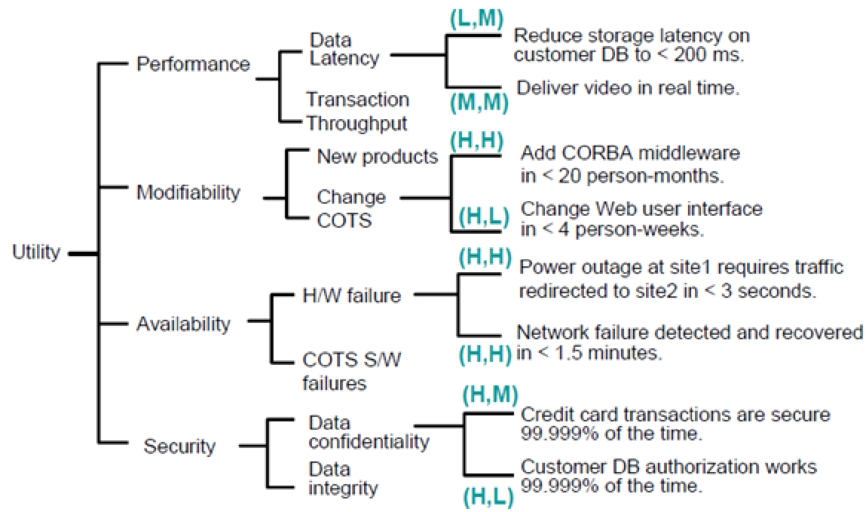


# Three Dimensions of Requirement Coverage

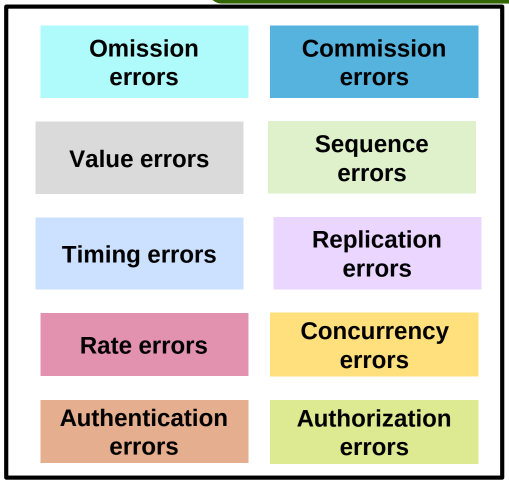
## System interactions, state, behavior



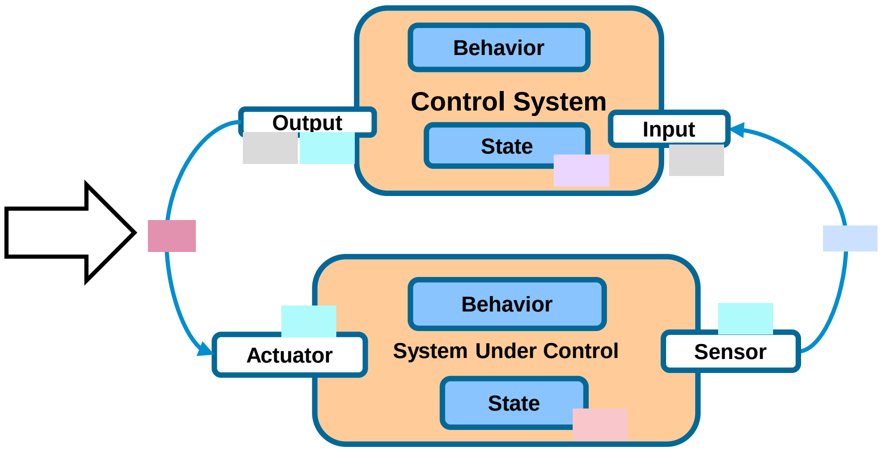

## Design & operational quality attributes




## Fault impact & contributors




Fault Propagation Ontology



# Impact and Alignment

DoD Acquisition and Industry Organizations

- OASD R&E: Champion maturation and insertion of virtual system integration into DoD programs
- DARPA research successes in HACMS program
- AMRDEC Joint Multi-Role (JMR) Tech Demo: maturation of Virtual System Integration for Future Vertical Lift (FVL) program
- Aerospace industry System Architecture Virtual Integration (SAVI) initiative Multi-year investment: Boeing, Airbus, Embraer, suppliers, FAA, NASA, DoD
- Rolls Royce engine control system case study

Standard Development

- Draft SAE AADL Requirement Specification standard
- Revision of SAE S18 ARP4761 System Safety Analysis standard

Regulatory Certification Agencies

- FDA: Guidance on medical device (re-)certification
- Underwriters Lab: medical device integration guidance (AAMI/UL2800)
- NRC: Educational workshop series on software system assurance

# Outline

Assurance challenges

Incremental life cycle assurance approach

**Year One Accomplishments**

# System Requirement Specification Notation

Associated with system specification in AADL architecture model

Description, rationale

Parameterization to accommodate variability changes

Traceability to stakeholder goals

Refinement, decomposition, hazard mitigation, evolution

Formal specification as predicate

Verifiable as specified in verification plans

Migration from textual requirement documents and traceability tools

```
system requirements FlightGuidanceImpl : "Requirements for the Flight Guidance System"
for FlightGuidance::FlightGuidance.subsystems
[
    val FG_DirectModeLatency =  0.15 ms
    val FG_NormalModesLatency =  25.0 ms
    compute ActualLatency
    requirement R6_1: "Stick-to-Surface End-to-End Flow Direct Mode Latency"
    for fStickToSurface_DirectMode
    [
        description "The stick-to-surface end-to-end flow response of the FGS in Direct Mode
            shall be no longer than " FG_DirectModeLatency
        value predicate ActualLatency <= FG_DirectModeLatency
        see document goal XACStakeholderGoals.XSG_8_3
    ]
```

# Verification and Assurance Plan Notation

Registry of reusable verification methods

- Verification precondition and result validation

Composable verification plans against system requirements

- Verification activities invoke parameterized methods on models
- Dependent and backup verification activities

```
claim R1 [
    activities
        actualsystemweight : Plugins.MassAnalysis()
        Weightlimit: mymethods.assumeWithWeightLimit()
        responsetime : Plugins.FlowLatencyAnalysis()
        behavior : Resolute.verifySCSReq1()
        timing: Plugins.ResourceAllocationScheduling()
    assert all [actualsystemweight else Weightlimit ,
        behavior , timing then responsetime
    ]
```

Configurable assurance plans

- Scope of assurance responsibility
- Time phased and priority focused execution of assurance plans

```
assurance plan AircraftTier2 for IntegratedAircraftSystem::AircraftSystem.
    assure own AircraftPlan
    assure subsystem plans FlightguidanceTb
    assume subsystems AircraftSubsystems::AuxiliaryPowerUnit
]

assurance task Tier2SafetyNetworkFocus for AircraftTier2 [
    filter verifications Network Safety only
```

# Assurance Automation and Metrics Collection

Assurance quality metrics

- Multi-valued verification result measures and their aggregates
  - Pass, fail, incomplete, conditions, backups
- Requirement coverage measures
- Weighted requirement claims, verification methods & results
  - Reflect importance, uncertainty, effectiveness

Measurement based assurance hotspot identification throughout lifecycle

- System AircraftTier2: (S93 F1 T0 E1 tbd0 EL0 TS0)
  - ClaimR1: The weight of the Aircraft system shall not exceed 70000.0kg (S2 F0 T0 E0 tbd0 EL0 TS0)
  - System ELE: (S1 F1 T0 E0 tbd0 EL0 TS0)
    - ClaimR1: The weight of the Electrical system shall not exceed 75.0kg (S1 F0 T0 E0 tbd0 EL0 TS0)
    - ClaimR2: The Electrical System shall be capable of handling at least 24000.0W (S0 F1 T0 E0 tbd0 EL0 TS0)
      - Evidence powercapacity: Analyze Electrical power demands against supply and capacity. This method is performe
        - system ELE: ** ELE power budget total 24500.0 W exceeds capacity 24000.0 W
        - system ELE: budget total 24500.0 W within supply 25000.0 W
  - System FGS: (S90 F0 T0 E1 tbd0 EL0 TS0)
    - ClaimR1: The FGS shall weigh no more than 300.0kg (S0 F0 T0 E1 tbd0 EL0 TS0)
      - Evidence weightlimit: Perform full weight (mass) analysis. This includes net/gross weight consistency, weight bud
        - system FGS: [G] Sum of weights 76.300 kg less than grossweight 280.000 kg (using gross weight)
        - system FGS: [A] Sum of weights 280.000 kg below weight limit 300.000 kg (6.7 % Weight slack)
    - ClaimR2: The FGS shall not draw in excess of 2000.0W (S1 F0 T0 E0 tbd0 EL0 TS0)
    - ClaimR3: The FGS shall be capable of processing at least 1100.0MIPS (S1 F0 T0 E0 tbd0 EL0 TS0)
    - ClaimR4_1: The RAM memory needs of the FGS shall be no more than 80 percent of 2048.0MByte (S1 F0 T0 E0 tb

# Requirement Spec & Virtual System Integration Case Study

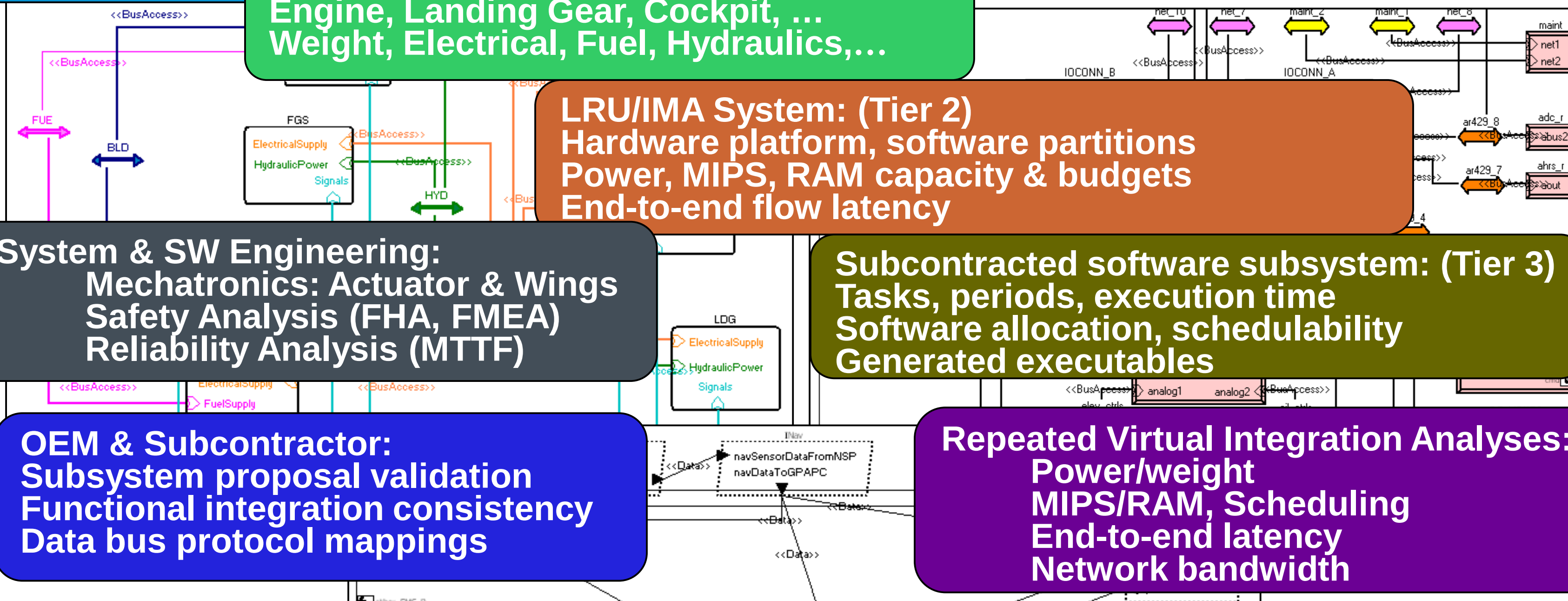
- **Joint Common Architecture (JCA) Demo**
  - Model based acquisition of FACE conformant software
  - Integration onto multiple Operating Environments

- **Architecture-Centric Virtual Integration Process (ACVIP)**
  - Shadow Effort to JCA Demo after BAA was released
  - By Software Engineering Institute (SEI), Adventium Labs, Software Engineering Directorate (SED)

- **Discovered potential system integration issues in advance through requirements, safety and timing analyses**
  - Early identification of 85+ potential integration issues

***Architecture analysis is critical for the successful and affordable integration of systems!***

# Assurance Rework Reduction through Virtual System Integration & Incremental Verification (SAVI)



**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**
- **Architecture-centric model-based acquisition and development process**
- **Multi notation, multi team model repository & standardized model interchange**

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

# Coming Up in Year Two

Focus on system quality improvement metrics

- Weighted requirement and verification result metrics
- Measurement-driven assurance hot spot guidance
- Demonstrate on three case studies

Demonstrate measurable requirement quality improvement

- Beyond stakeholder traceability, “shall,” no “not”
- Improvement of three coverage dimensions

Demonstrate measurable verification cost reduction

- Cost reduction through automated safety analysis
- Uncertainty reduction through priority focused architecture design exploration
- Proportional recertification costs through compositional automated verification

# Contact Information

**Peter Feiler**

Software Solutions Division

Email:  phf@sei.cmu.edu

**U.S. Mail**

Software Engineering Institute

Customer Relations

4500 Fifth Avenue

Pittsburgh, PA 15213-2612

USA

**Customer Relations**

Email: info@sei.cmu.edu

SEI Phone: +1 412-268-5800

SEI Fax: +1 412-268-6257