Software Interaction Complexity Drives System Cost

Software Development Life Cycle

Where Faults are Introduced

- Requirements: 70%
- Architecture Design: 20%
- Code: 10%
- Unit Test: 3.5%
- Integration Test: 16%
- Acceptance Test: 50.5%
- Operation: 9%

Where Faults are Found

- Nominal Cost Per Fault for Fault Removal: 20.5%
- Cost Per Fault for Fault Removal: 300–1000x

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

Post-unit test software rework cost 50% of total system cost and growing

Software as % of total system cost
1997: 45% → 2010: 66% → 2024: 88%
The AVSI SAVI Program

System Architecture Virtual Integration (SAVI)

- Aerospace Vehicle Systems Institute (AVSI) is a consortium of commercial aerospace companies and gov’t agencies
- AVSI launched SAVI in 2008 to address the problem of growth in complexity of systems leading to cost and schedule overruns
- The objective is to develop a standards-based Virtual Integration Process (VIP) that allows multiple parties to virtually integrate and analyze systems throughout development life cycle
- The result is earlier detection and correction of errors leading to cost savings
- Highly focused on integration – defining the state of the art in system integration consistency checking
AADL focuses on interaction between the three major elements of a software-intensive system based on architectural abstractions of each.
• Requires standardized architecture modeling language with well-defined semantics integrating hardware, software and systems
• Includes virtual, incremental, hierarchical, compositional analysis of a software-reliant system to evaluate integration effects
• Avoids the perpetuation of requirements defects into later phases of the development process enabling major rework cost reduction
• Increases assurance confidence by augmenting testing
• Enables rapid generative integration of the verified system
• Includes a “Single Source of Truth”
• Leverages the AVSI SAVI Project

Virtual Analysis & Integration of Software, Hardware, and Systems
# Joint Multi Role (JMR) Mission Systems Architecture Demonstration (MSAD) Schedule

## Joint Multi Role (JMR) Mission Systems Architecture Demonstration (MSAD) Schedule

### Joint Common Architecture (JCA) v1.0 Development

**Tasks**
- Assimilate MS ETA Results
- Coordinate with Community
- Compile Supporting Docs
- Functional Decomposition
- Mission Set Allocation
- Semi-annual Updates

**Products**
- Functional Model
- Data Model
- Guidance Documentation
- JCA Tools

### JCA Demo / ACVIP Shadow

**Tasks**
- Scope limited to single component
- Model Based Acquisition
- AADL Modeling / Analysis
- JCA Model Refinement
- Lab Integration / Testing
- Report Generation
- Process Refinement

### Architecture Centric Virtual Integration Process (ACVIP)

**Tasks**
- Assimilate MS ETA Results
- Coordinate with Community
- Compile Supporting Docs
- Functional Decomposition
- Mission Set Allocation
- Semi-annual Updates

**Products**
- Analysis Tool Maturation
- Tool Documentation
- Model Translators / Interfaces
- Demonstration Models
- Process Definition
- Process Maturation

### Objective MEP Definition

**Approach**
- Apply System-Theoretic Process Analysis (STPA) to FVL CONOPS, JCA and MEP Definition
- Under MIT / Lincoln Labs Leadership

### Safety / Security Study

**Approach**
- Apply System-Theoretic Process Analysis (STPA) to FVL CONOPS, JCA and MEP Definition
- Under MIT / Lincoln Labs Leadership

### Mission Systems Architecture Capstone Demonstration

**Approach**
- Specification for a full mission systems architecture
- Multiple vendors
- Model Based Acquisition
- JCA / FACE Validation
- Scope of implementation limited by available resources (i.e. design only, limited lab implementation / test, etc.)

### Architecture Implementation Process Demonstrations (AIPD)

**Approach**
- Government defined areas of emphasis and goals related to JCA, FACE™, ACVIP and MBE in general
- Efforts provide “evidence” of ability to meet USG business and process goals and are relevant to industry and Army aviation PM plans
- High level of collaboration between USG and industry
ACVIP requires the processes and tools to be established and matured for effective use on Future Vertical Lift (FVL).

ACVIP will be exercised, documented, applied and enhanced throughout the Mission System Architecture Demonstrations:
- Definition, development and exercise of the ACVIP tools is initially performed by the tool developers.
- Application and evaluation is conducted during MSAD demos supporting maturation and technology transition.

**ACVIP Process & Tool Maturation**
1. Define
2. Develop
3. Exercise
4. Document
5. Apply
6. Evaluate

**Near Term MSAD Tasks related to ACVIP**
- Develop First Edition ACVIP Handbooks
- AADL/ACVIP Training
- Provide tools for use during AIPD
ACVIP as part of a Model Based Acquisition approach
- Model requirements in AADL and perform ACVIP analyses prior to solicitation for initial system architecture.
- Systems architecture requirements specification model in solicitation and response.
- Systems integrator communicates with component vendors via models.

Government’s role in the development and analysis process
- Government sponsors independent analyses using architecture models.
- Government requires receipt of architectural models in AADL and model ownership.
- ACVIP analyses performed by contractor and Government during each architectural phase (functional, conceptual, design, integration).

Single source of truth
- Models will be used throughout the life-cycle and updated over time.
- Will be used by multiple technical domains when performing analyses.
- Models must be accurate, up-to-date, phase appropriate and integrated.
- Distributed model repository for Government/integrator/supplier collaboration.
- AADL used as the standard method of communication between tools and across organizations.

Communication of accurate information and virtual analysis is key
What is FACE™?

- Future Airborne Capability Environment (FACE™)
- Open standard established by DoD and Industry via The OpenGroup©
- The FACE™ architecture comprises points where variance occurs (i.e., layered architectural segments)
- A SOFTWARE computing environment to enable product lines for military aviation
- Eliminates barriers to software portability, prevents lock-in and improves competition
- Not only a technical standard but also includes a business strategy
- Includes:
  - Development Ecosystem
  - Conformance Test Suite
  - Verification & Certification
  - Repository

Learn more @ http://www.opengroup.org/face/face101
What is Joint Common Architecture (JCA)?

- JCA is a Reference Architecture (not a system architecture) for FVL Family of Systems

- JCA Guides and constrains architecture implementations by providing:
  - a common lexicon and taxonomy
  - a common (architectural) vision
  - modularization and the complementary context

- JCA v1.0 describes conceptual avionics capabilities with specific focus on the Mission Computer (MC) subsystem

- JCA includes:
  - Functional Model
    - Decomposed Mission Level Capabilities allocated to the MC subsystem and their top level organization and interactions
  - Semantic Model
    - Conceptual level
    - Linked to Functional Model
  - Model Analysis
    - Model representation in AADL allowing ACVIP type analysis
  - Documentation
    - Development Plan
    - Implementation Plan
  - Tools/Ecosystem
    - Translation of the JCA v1.0 conceptual model into FACE v3.x conformant conceptual and logical models
    - JCA conformance
Modular Integrated Survivability (MIS) System

Goals/Objectives
- Validate the JCA & FACE approaches
- Mature JCA, FACE Standard & Ecosystem tools, and business practices
- Gain experience implementing a model based approach (learn by doing)

Approach – Controlled Experiment
- Procure single software component from multiple vendors built to same specification
- Integrate component on two undisclosed Operating Environments (OEs)
- Execute a representative model-based acquisition approach
- Limit developers and integrator interactions
- Use FACE Ecosystem tools
- Exercise FACE Verification Authority process
- Procure a Reusable Verification Component
- Conduct a partial ACVIP as a Shadow Effort

Payoff
- Reduces risk for future implementations
- Provide Government greater insight than from a typical acquisition

JCA Demo Schedule

<table>
<thead>
<tr>
<th></th>
<th>FY 13</th>
<th>FY 14</th>
<th>FY 15</th>
</tr>
</thead>
<tbody>
<tr>
<td>Solicitation</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RFI</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BAA</td>
<td></td>
<td></td>
<td>G</td>
</tr>
<tr>
<td>Award</td>
<td></td>
<td></td>
<td>B</td>
</tr>
<tr>
<td>SW Development</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lab Integration</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Conformance &amp;</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Integration Test</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Demo/Report</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
ACVIP Shadow Effort was performed by independent analysts from the Software Engineering Institute (SEI) of Carnegie Mellon University and Adventium Labs with information obtained from the component specification, component developers and system integrator. The purpose of the Shadow Effort was to:

- Gain experience with ACVIP
- Demonstrate potential of ACVIP
- Maturate Government-sponsored AADL tools

ACVIP was effective in the conduct of post-award analysis for:

- Requirements – checked for consistency, accuracy, completeness
- Safety – identified potential safety hazards and safety assessment
- Timing – analyzed adequacy of resource scheduling and partitioning

ACVIP Shadow effort did not demonstrate a representative ACVIP process; however, significant experience and results were obtained.

- Pre-solicitation requirements analysis would have reduced errors.
- Models were not reused or integrated in a repository.
- Analytical tools and processes were very immature.
- Lack of accurate and/or complete information from DCFM developers and system integrator. Many assumptions had to be made.
ACVIP Process on JCA Demo

- **AADL Model Construction in OSATE**
  - Assumptions
  - Conceptual Integration Model
  - Architecture Led Requirements Specification (ALRS) Analysis
  - Architecture Led Safety Analysis (ALSA)
  - Architecture Led Timing Analysis

**Stakeholder Requirements**
- MIS Stakeholder Requirements
- MIS System Requirements
- MIS Stakeholder Requirements
- MIS Stakeholder Requirements

**Modeling Tools**
- DCFM Supplemental Requirements
- DCFM EA UML Data Model
- MIS Rhapsody UML Model
- MIS Build 2 Plan (system description)

**Analysis Reports**
- Requirements Analysis Report & Model
- Safety Analysis Report & Model
- Timing Analysis Report & Model
• Textual requirements results in:
  − Ambiguous, missing, incomplete and inconsistent requirements.
  − Cost and schedule impacts due to error injected in the design

• Solution:
  − Represent verifiable requirements in an architecture model (based on AADL RDAL* Annex)

• ALRS Analysis Process:
  − Every element of a system specification must be addressed by requirements
  − Non-functional requirements are driven by utility trees as output of an ATAM**
  − Resulting annotated model is basis for Architecture-led Safety Analysis (ALSA)

* RDAL = Requirements Definition & Analysis Language
** ATAM = Architecture Tradeoff Analysis Method™
Aircraft Survivability Situational Awareness System (ASSA) = DCFM Integrated with the MIS

Collection, Correlation, Fusion, Assessment of observations => Inform and Act

Standardized data formats?

AS-SA Health Monitor & Configuration: MIS subsystem?

Max response latency < 1.6 sec

Streams of time-stamped observation track sets

SA Data Conversion & SA Data Service as two MIS subsystems

DCFM:
Single or multiple DCFM instances?
Include threats only or also terrain, own and adjacent aircraft?
• SEI conducted a safety analysis of the JCA Demo system using ALSA

• ALSA based on the AADL Standard Error Model Ver 2 (EMV2) Annex

• EMV2 supports SAE ARP 4761

• An error propagation ontology guides identification of hazards

• Hazard annotations of AADL model lead to automated fault impact analysis and report generation

• ALSA extendible to security
- JCA Demo BAA assigned Design Assurance Level (DAL) E to DCFM, but:
  - Aircraft are lost to operational threats, obstacles, and terrain.
  - Multiple minor hazard contributors can have catastrophic consequences.
  - Embedded software as major hazard source: unexpected interaction behavior.
- SEI demonstrated the use of ALSA to assist in identifying the appropriate DAL for the system
  - Identified critical areas related to the reporting of false positives, false negatives, untimely information.
  - Derived Health Monitoring System (HMS) requirements, clarifying expected functionality of the HMS.
  - Safety hazards introduced by health monitor.
• Challenge: analysis of end-to-end timing for distributed, multidisciplinary, heterogeneous computer systems
  - Different scheduling on different network and processing nodes
  - Co-existence of sampled and event-driven processing of time sensitive information

• Two approaches: simulation and schedulability analysis
  - ACVIP Shadow focused on schedulability analysis
  - Survey identified 16 schedulability analysis tools

• Adventium developed Framework of Schedulability, Timing and Resources (FASTAR)
  - Integration of variable scheduled subsystems and end-to-end analysis

• For JCA Demo FASTAR integrated the following two timing analysis tools:
  - MAST: Modeling & Analysis Suite for Real-Time Systems for Switched Networks analysis
  - SPICA: Separation Platform for Integrating Complex Avionics for Partitioning analysis
- Functional Architecture Mapped to Hardware Architecture
  - Identified system timing issues in layered hardware architecture
  - Hardware with different clock domains (e.g., Switched Ethernet & ARINC 653)
  - Unsynchronized compute module and the remote sensors
  - Mixed fidelity modeling and analysis
• Unclear requirements between component states and MIS system state
• Missing specification of currency/staleness for the data and end-to-end timing requirement for hazard data
• Partition schedule did not meet ARINC 653 scheduling rules
• Non-clarity in protocol from MIS to support single or multiple DCFM instantiations
• Absence of data storage requirements between the DCFM and MIS
• Ambiguity of MIS system Operational State when a clock timer expires
• Missing source track quantity requirements for the aircraft survivability sensor
• Potential for track jitter
• Multiple sensor stream rates could create potential issues and requires further analysis
• Disagreement in threat thresholds between the DCFM and MIS
• Lack of memory requirements in MIS that could lead to memory leaks
• Ambiguity in the requirement to correlate 50 source tracks within 1 second

86 potential issues were elicited by ACVIP
• Process of modeling the architecture alone can uncover requirement issues
• Value in performing Architecture Led Requirement Specification analysis
• Accurate system information is required for valid analysis
• Translation between AADL and other modeling languages would reduce ambiguity and improve communication
• Metrics for measuring ACVIP effectiveness are lacking
• Additional timing analysis tools need to be added to FASTAR to address a broader range of timing issues.
• AADL training proved beneficial
  – Provided government personnel with insight into AADL modeling
  – Created interest with industry
Conclusions from JCA Demo ACVIP Shadow Effort

- Performing ACVIP analysis prior to the release of BAA would have been beneficial to overall program execution.
- ACVIP analyses could reduce error perpetuation from requirements phase to system integration & test.
- Many of the ACVIP tools are currently immature.
- Demonstrations can identify, validate, mature and transition methods and tools to support ACVIP.
- ACVIP must transition from execution by tool developers to Government and industry personnel.
- A full ACVIP needs to be exercised in a demonstration in order to understand the complete value of ACVIP.

JCA Demo ACVIP Shadow was successful in providing the Government with experience and validating the ACVIP concept.
Alex Boydston is an electronics engineer working mission systems and architecture demonstrations for the US Army Aviation Development Directorate at Redstone Arsenal for the Joint Multi Role science and technology program to support the joint services Future Vertical Lift initiative. Alex has a Bachelor of Science and a Master of Science degrees in electrical engineering from the University of Alabama in Huntsville. Alex has over 25 years of engineering experience working for organizations such as Teledyne Brown Engineering as a communications systems engineer on the National Missile Defense program, payload systems integrator for NASA shuttle and station programs, AdTran Corporation as an embedded systems product design and test engineer, and Draper Laboratory and the U.S. Army as an avionics engineer.