Avionics Software Challenges and Initiatives

Briefing to Software Design and Productivity Workshop
18-19 April, 2001

Don C. Winter
Manager – Open Systems R&D Programs
Phantom Works
Avionics Software Situation

Software has become the pacing element in the development and modernization of military avionics systems

- Accounts for a growing percentage of system complexity and cost
- Increasing instances of cost/schedule overruns
- More errors slipping through to fielded systems

The Challenge: Provide affordable, maintainable, high-integrity software within budget and delivery constraints
Problem Statement

• Technology trends in weapon systems are driving exponential growth in software complexity
  – Autonomous systems, adaptive systems, fault-tolerant systems...
• Traditional approaches and processes do not scale well
  – Program-specific architectures, languages, tools
  – Unaligned with commercial practices
• High turnover in defense software workforce
  – Ad-hoc knowledge management for legacy systems
  – Constantly climbing the learning curve

Current technology, practices and culture of the industry cannot cope with the emerging environment
Product Line Software Architecture

- Hierarchical layered architecture
- Isolation from hardware changes
- Plug-and-play software modularity
- Reuse applications
- Change encapsulation
- Improved maintainability
- Improved productivity

Common OFP architecture

Application software

Infrastructure (COTS)

Hardware (COTS)
Where are the Technology Voids?

COTS supports some aspects of development well, but many voids exist:

• “Front end” of process
  – Model-based tools for requirements/design capture
  – Automated configuration and integration of components

• “Back end” of process
  – Simulation-based testing
  – Formal verification methods and tools

• Support for hard real-time, embedded systems is limited
  – Quality-of-service requirements expression/guarantees

• Support for high-dependability systems is limited

• Legacy system constraints
  – Infusing new technology into resource-limited, “closed” systems
Cultural Challenges

• Defense acquisition culture presents impediments
  – “Silo” approach to planning/funding system modernization
  – “Wasn’t invented here” mindset in programs
  – Inability to trade front-end investment for life-cycle returns, even when business case is compelling
  – Support structure based on single fielded configuration
  – T&E community resistance to tailored re-qualification
• FAA software development/verification culture presents additional impediments
  – Approved processes lag technology
  – “Cradle to grave” involvement and oversight
• Synergy with COTS industry will always be limited without cultural transformation
Summary

• Boeing remains fully committed to a product line strategy for avionics software
  – Founded in COTS, open systems architecture
  – Large, multi-year IRAD investment
  – Growing number of platforms
• Government/industry partnership is needed to realize the full potential
  – Complementary investments in technology
  – Acquisition reform
  – FAA/certification authority involvement
To leverage COTS technology, military system developers must establish boundaries of stability:
- System functionality must remain stable for many years
- Changes forced by obsolescence must be contained
  > Re-qualification is too expensive to do often
- Sources of COTS are not under configuration control
## Board Support Package
- Tailors the OS for Hardware Platform
- COTS Software (VME Drivers, Interrupt Handlers)

## Operating System
- COTS Real Time Operating System (e.g. VxWorks from Wind River)
- POSIX Compliant Run Time Environment

## Infrastructure Services / API
- Common Object Request Broker Architecture (CORBA) Compliant Infrastructure
- Facilitates Distributed Object Oriented Computing and Communication

## Application Software
- Object Oriented Design Methodology
- Written in C++/Ada High Order Languages
- Contains Reusable S/W "Building Blocks"
## COTS Leverage - Development Tools

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Requirements Analysis and Management</td>
<td>• DOORS® Toolset for Requirements Capture</td>
</tr>
<tr>
<td>Software Development</td>
<td>• Rational Rose® Toolset for Object Oriented Analysis / Design</td>
</tr>
<tr>
<td></td>
<td>• Microsoft Visual C++ for Program/Debug</td>
</tr>
<tr>
<td>Auto Code Generation</td>
<td>• VAPS® Toolset for Open GL Prototype Display</td>
</tr>
<tr>
<td></td>
<td>• Rational Rose Tool for C++ Code Generation</td>
</tr>
<tr>
<td>Desktop Test Environment (DTE)</td>
<td>• Facilitates Early Testing of S/W Components</td>
</tr>
<tr>
<td></td>
<td>• Reduces Requirements for S/W Test Benches</td>
</tr>
<tr>
<td></td>
<td>• Contains Microsoft Visual C++ Code Debugger</td>
</tr>
<tr>
<td></td>
<td>• COTS Graphical User Interface Developer</td>
</tr>
<tr>
<td></td>
<td>• VAPS® Display Simulation</td>
</tr>
<tr>
<td></td>
<td>• Common Test Language</td>
</tr>
<tr>
<td></td>
<td>• C++ Aircraft/Avionics Simulation</td>
</tr>
<tr>
<td>Configuration Management</td>
<td>• ClearCase / ClearQuest</td>
</tr>
</tbody>
</table>
Early Returns - Measured Benefit

Cumulative Software Development Productivity

Key Sources of Gain:
- Reuse
- COTS Tools
- Change Containment
- Desktop Testing

Historical F/A-18 Legacy Cost

Historical F-15 Legacy Cost:

Goal

F-15
F/A-18

Demonstrated Software Development Cost Reduction to less than 25% of Legacy Programs
Where to Look First - the Commercial Sector

COTS market appears to be getting the job done, but:
- System turnover rates are much higher
- Systems are fielded with bugs
- Deadline-driven culture vs. “getting it exactly right” culture
- Systems are not as capacity-limited
  Inherently greater leverage of automation and re-use

Product Cycle - Military Vs Commercial

Product Lifetimes are getting Shorter