The Modernization of the Aegis Fleet with Open Architecture

Andrew Winkler

Sept 28, 2011
Topics
Things to talk about...

• **Background**
  – Aegis Overview
  – Capability Upgrade Evolution
  – Modernization Concept/Approach

• **Aegis Open Architecture**
  – Evolution to COTS Technologies and Products
  – Incremental/Spiral Development Approach

• **Aegis Modernization**
  – Overall Scope/Impact
  – Product Line Architecture
  – Integration of Common STM / TS Components

• **Summary**
Aegis – The Shield of the Fleet

Design Cornerstones
- Continuous Availability
- Surveillance Coverage
- Reaction Time
- Firepower
- Environmental Resistance

ASCM High Diver

ASCM Sea Skim

Land Launch ASCM

CVN

DDG

Self Defense | Area Air Defense | Long Range Air Defense And BMD
Aegis Combat Systems Architecture

- Aegis Display System
- Radar System AN/SPY-1
- Command and Decision System
- Vertical Launching System Mark 41
- Fire Control System Mark 99
- Weapon Control System
- Aegis Combat Training System Mark 50
- Operational Readiness Test System
- Standard Missile-2 SM-3
- VLA TLAM
- ESSM
Aegis Capability Overview

12 Generations and Over 27 Years of Proven Success
Aegis Modernization Concept

I. Decouple Hardware and Software Upgrades Using COTS
   - Software Upgrades Every Two Years
   - Hardware Refresh Every Four Years

II. Build on Fielded Baselines
III. Integrate Navy Enterprise HW and SW Solutions
IV. Transition Aegis to Navy Objective Architecture

Benefits of Aegis Modernization Concept
   - More Capability to the Fleet Sooner
   - Foster Collaboration and Competition
   - Cost Savings from Commonality & Reuse
   - Minimal Lifetime Spares
   - Upgrades Backward Compatible

COTS & Open Architecture - While Maintaining Engineering Discipline
Aegis Open Architecture
## AWS Computer Architecture Evolution

|-----------|------|------|------|------|------|------|

### Processors
- **Processors**
  - UYK-7
  - UYK-20
  - UYK-43
  - UYK-44
  - UYK-43/44+ Adjunct COTS

### Software
- **Software**
  - CMS-2
  - C++, Ada

### Interfaces
- **Interfaces**
  - NTDS Parallel
  - NTDS Serial
  - NTDS Parallel FDDI / Ethernet

### Displays
- **Displays**
  - UYH-4
  - UYQ-21 (TGC*)
  - UYQ-21/UYQ-70

### System Complexities
- **UYK-43s B5PllII TGC**
- **SMP’s ALIS Network**

### Notes
- * Applicable to Baseline 5 Phase III Only
- ** Eliminated in Baseline 7 Phase IR

---

Copyright © 2011 by Lockheed Martin Corporation
COTS Technology and Products

**Tech Insertion 00** 2000  
**Tech Insertion 04** 2004  
**Tech Insertion 08** 2008

- B7Phl  
  - DDG 91-102
- B7PhlIR  
  - DDG 103-112  
  - LCS / NCS - Derivative
- ACB 08  
  - CG 52-59

**Non-LM Hardware**

- Computing Platform
- VME Single Board Computer
- Network Switching
- SAN Storage
- Network File System
- Thin Client LCD Display
- Analog Hardware/Devices

**Non-LM Software**

- Real Time Operating System
- Pub-Sub Communications
- High Availability Middleware
- Enterprise System Management
- Human-Systems Software
- Network Management Tools

**Smaller Footprint and Reduced Processor Costs**
Incremental Development
“Build a little ... test a lot”

- Spiral-1
  - OA Radar
  - At-Sea Demo
  - DDG 96

- Spiral-2
  - OA Display
  - OA Weapons
  - WSMR

- Spiral-3
  - DDG 103-112

- Tech Insertion 08
  - ACB 08
  - CG 52-59

- Completed Modular Designs
  - 2007

- Fielding Modular Design foundation
  - 2008-2011

Open Architecture Foundation for Baseline 9 Developments

• Focused on Radar, Weapons
• Model-Centric Development (UML)
• Modern Languages (C/C++, Java)
• Non-Proprietary Interfaces
**Where We are Today**

Aegis Combat System

Aegis LAN Interconnect System Common Network (ALIS III)

<table>
<thead>
<tr>
<th>Technical Assessment</th>
<th>SPY</th>
<th>Open C2</th>
<th>Weapons/Fire Control</th>
<th>VLS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Signal Processing</td>
<td>Radar Control</td>
<td>Display</td>
<td>C&amp;D</td>
</tr>
<tr>
<td>DDG-91 2004</td>
<td>Hardware Custom SMP</td>
<td>Mainstream SMP</td>
<td>Mainstream SMP</td>
<td>Mainstream</td>
</tr>
<tr>
<td></td>
<td>Software Closed SMP</td>
<td>Open SMP</td>
<td>Open SMP</td>
<td>Closed SMP</td>
</tr>
<tr>
<td>CG-52 2008</td>
<td>Hardware Custom Mainstream</td>
<td>Mainstream Mainstream</td>
<td>Mainstream Mainstream</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Software Closed Mainstream</td>
<td>Open Mainstream</td>
<td>Open Mainstream</td>
<td>Closed Mainstream</td>
</tr>
<tr>
<td>CG-62/DDG-51 2012</td>
<td>Hardware Mainstream Mainstream</td>
<td>Mainstream Mainstream</td>
<td>Mainstream Mainstream</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Software Open Mainstream</td>
<td>Open Mainstream</td>
<td>Open Mainstream</td>
<td>Open Mainstream</td>
</tr>
</tbody>
</table>

**Continuously Advancing the Aegis Combat System Forward**

Copyright © 2011 by Lockheed Martin Corporation
Today’s Aegis Combat System
Surface Warfighting Electronics Architecture

Detect/Control/Engage View

System/Subsystem View

- Federated, Tiered Architecture
- Efficient ACS Capability changes
- Well-Define Components and API’s

Aegis LAN Interconnect System Common Network (ALIS III)

SPY-1 Radar Sig Pro C&D Display Weapons VLS

Track Management Doctrine Air Control Weapons

Supports Operational and Navy Business Model Objectives

Copyright © 2011 by Lockheed Martin Corporation
Aegis Modernization
Roadmap to Aegis Modernization (AMOD)

**AMOD ACB12 (TI12)**
- Aegis BMD 5.0
- ACS Element Upgrades
- NIFC-CA
- JTM Alignment
- MMSP
- SM-6

**AMOD Advanced Capability Build 12 (DDG configuration)**
- Tech Insertion (TI) 12
- Aegis BMD 5.0
- ACS Element Upgrades
- NIFC-CA
- JTM Alignment
- MMSP
- SM-6

**AMOD ACB08 (TI08)**
- OA Spiral 3
- ACS Element Upgrades
- TI 08

**Aegis BMD 4.0.1**
- Improved Discrimination
- Improved Track Handover
- Enhanced LoT
- Integrated IR/RF KA
- SM-3 Blk IA and IB

**Aegis BMD Block 06/08**
- LRS&T, Engagement and LoT
- Multi-Mission
- Integrated Mission Planning
- SM-3 Blk I and IA

**Aegis BMD Block 04**
- COTS architecture

**CR0/CR1**
- B/L 7 Phase I
- COTS architecture

**Increased Battlespace and Multi-Mission Interoperability**

**COTS Based Infrastructure**
AMOD Technical Scope

Aegis Weapon System
COTS Refresh 3

Aegis Combat System Upgrades

HM&E Upgrades

Sea-Based Ballistic Missile Defense

BMD 4.0.1 Functionality and SM-3

NIFC-CA and SM-6

JTM Alignment

Copyright © 2011 by Lockheed Martin Corporation
Way Ahead … Baseline 9

Focus: SPY, Weapons, and Display

“ACB 08”

Fielding Modular Software
• Open Standards
• Collaborative Peer Review Environment
• Full Government Purpose Data Rights
• Leveraging small business innovation
• Established Two Technology Centers

Modular Design Foundation

Steering and Priming The SBIR Process

USS Bunker Hill CG-52

Integrate 3rd Party

Applying S&T Investments Capability Based Plan – Baseline 9 Aligned

Balancing Capabilities with Complex Combat System Integration Foundation Established for Transition to Objective Architecture
Implementing Open Architecture
Layered Architecture Foundation

Infrastructure:
- Common Services and APIs
- Flexibility to Support Forward-Fit and Back-Fit

Common Computing Environment:
- Standards-based Interfaces to network
- Commercial Mainstream Products and Technologies

Componentized Objective Architecture:
- Common Reusable Components
- Platform Specific Components
- Data Model
- Extensible to the Future

Decouple Hardware (H/W) from Software (S/W)

Upgrading Hardware and Software Independently
Joint Track Management Alignment

Overview

• Align AMOD and SSDS Track Management to a Common Architecture
  – Provide Consistent Functional Allocation, Data Representation and Attributes
  – Incorporate Reusable System Track Manager and Track Server Components

• Provides Hierarchical Track File (System Level – Source Level)

• Provides Standard Interfaces
  – Track Server Standard Access Interface for Client Applications
  – Track Manager Integrates Track Data Sources via Common Interface; Extensible for New Track Data Sources

• Provides Two Complete Versions of Live Training Tracks:
  – Allows Training Override of Multiple Attributes
  – Training Tracks Can be Physically Relocated From Live Location

• Provides Dual Ownership – Tactical and Training:
  – Allows Training View to be Repositioned with No Impact to Tactical View

Aligning the Architecture for Future: Common Components Across Ship Classes
SI/DA Scope

- Implement Common Track Server
  - Replace Existing PDM Component
  - Implement configurable track server – support multiple track sources
  - Standardize services and APIs
  - Integrate PDM, DDG-1000 and SSDS track server design concepts
  - Ensure resulting component will work for both AMOD and SSDS

- Consolidate System Track Manager
  - Implement JTM hierarchical design approach
  - Consolidate STM and MNF
  - Improve track file and data xfer (e.g., capacity, types, attributes, …)
Component Framework Services

Key:
- **Green:** Common Components
- **Yellow:** Component Framework Services
- **Pink:** Aegis C2/System Services
- **Gray:** COTS

Future Common Components

STM

TS

Component Framework Services

Availability Management

Subcomponent Management

Navigation

Operational Modes

Data Recording

Logging

P/S Messaging

Time

C2 Libraries

DXR

RTI DDS

System Management (OASM)

GA SelfReliant

Operating System

Hardware

Copyright © 2011 by Lockheed Martin Corporation
Common STM and TS Components

Task Allocation...

**LM Tasks:**
- Update AMOD System Specs (A-level, B1, B5)
- Provide Legacy Aegis Requirements (e.g., STM, PDM, MNF) to SI/DA
- Validate Aegis Requirements Covered by Enterprise SRS’s
- Remove STM/TS Functionality from Existing Components
- Modify C&D Sensor Managers IAW Functional Allocation (Design, Code, and Test)
- Modify Aegis Track Server Clients (Design, Code and Test)
- Design, Code and Test Aegis-Specific Component Framework
- Integrate STM/TS into AMOD
- Provide TOR/CPCRs
- Verify System Performance

**SI/DA Tasks:**
- Develop Enterprise SRSs for STM and TS from Aegis and SSDS
- Develop UML Models
- Auto-generate IDD and Interface Code from UML Models
- Design, Code and Test STM and TS Components
- Provide Interim and Final STM/TS Components to LM
- Implement CM and Change Control of STM/TS
- Implement CPCR Fixes to STM/TS Components
- Support Integration of STM/TS into AMOD
- Support SQT of STM and TS

**LM and Third party Joint Tasks:**
- Establish linked classified development environment
- Establish and Track Progress and Dependencies via Joint IMS
- Participate in Navy-led Data Model and Component Framework Working Groups
- Support Functional Allocation
- Support Definition of Data Model, TS APIs, and Common Service APIs
- Support Definition of Enterprise-level Processes and Artifacts
- Support Enterprise ETRs and Enterprise SSR
- Support Enterprise CCB and Prioritization/Adjudication of TORs/CPCRs

**Legend**
- New/Modified AWS
- New Common

Allocation and Governance Was Essential
Objective Architecture
Roles and Responsibilities...

Product Line Development

ETR
- Peer Review
- Comment Adjudications

CM
- Cross-Program Change Control
- Cross-Program Review Boards
- Decisions

Govern
- Style and Format

Templates

ADD
- Perform System Modeling
- Allocate Requirements and Performance Budgets
- Develop System Specs
- Flow down Reqmts to Software Components

Component SRS
- Architecture Precepts / Patterns
- Component Responsibilities / Interfaces
- Functional Allocation
- System Use Cases / Threads
- Common Data Groups

Develop Components

Component

Track Data Model

Message Definitions

TS APIs

Infra SVCs APIs

Vehicle Control DM

Sensor Mgmt DM

SSDS

Aegis

Analyze/Define System
- Perform System Modeling
- Allocate Requirements and Performance Budgets
- Develop System Specs
- Flow down Reqmts to Software Components

Integrate Components
- Support Developer Peer Reviews
- Integrate into System Configuration

Verify System
- Support T&E through Sell-off

Copyright © 2011 by Lockheed Martin Corporation
AMOD C&D Component Architecture
Message Processing...

Source/Comms Management
- MTF
- JTF
- SPF
- GTF
- LEF
- QBF
- RCF
- CEF
- LIF
- IAF
- STM
- DIP
- BMF
- SPQ-9B (CG Only)
- SPS-67 (DDG Only)

System Track Management
- DIF
- TS
- DDF
- CLM
- DXR
- IPC
- CSM
- NVF
- CEF
- LEF
- SPQ-9B (CG Only)
- SPS-67 (DDG Only)

Display Management
- HIF
- EMF
- EFF
- DMF
- ICF
- DCF
- MSF

Engagement Management
- Engagement Management
- HWS
- GWS
- WCS
- ASW

C2 Services
- MTF
- JTF
- SPF
- GTF
- LEF
- QBF
- RCF
- CEF
- LIF
- IAF
- STM
- DIP
- BMF
- SPQ-9B (CG Only)
- SPS-67 (DDG Only)

Planning
- MTF
- JTF
- SPF
- GTF
- LEF
- QBF
- RCF
- CEF
- LIF
- IAF
- STM
- DIP
- BMF
- SPQ-9B (CG Only)
- SPS-67 (DDG Only)

Identification
- MTF
- JTF
- SPF
- GTF
- LEF
- QBF
- RCF
- CEF
- LIF
- IAF
- STM
- DIP
- BMF
- SPQ-9B (CG Only)
- SPS-67 (DDG Only)

Resource Management
- MTF
- JTF
- SPF
- GTF
- LEF
- QBF
- RCF
- CEF
- LIF
- IAF
- STM
- DIP
- BMF
- SPQ-9B (CG Only)
- SPS-67 (DDG Only)

Common Messages Across Ship Classes
One IWS Track Data Model: ~130 Messages
What We Learned

Lessons Learned Address Multiple Perspectives
Aegis Open Architecture

Summary

1994

- COTS Infrastructure
  - Separation of Application/Infrastructure
  - Commercial Standards
  - Commodity Products

2000

- B7Phl
  - DDG-91+

Component-Based Software

- Component-Based Designs
- Layered Architecture
- Configurable Test Environments

2006

- CGM
  - CG-52+

Open Business/Common Components

- Objective Architecture
- Open Business Practices
- Open Disclosure / Gov’t Purpose Data Rights
- Increase Number of Players/Opportunities

2012-2016

- AMOD
  - DDG-51+/DDG 113

- Increased Capabilities
  - AAW/BMD
  - JTM
  - SM-6
  - NIFC-CA
  - SBT

Copyright © 2011 by Lockheed Martin Corporation
## Glossary

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACB08</td>
<td>Advanced Capability Baseline 2008</td>
<td>LAN</td>
<td>Local Area Network</td>
</tr>
<tr>
<td>ACB12</td>
<td>Advanced Capability Baseline 2012</td>
<td>LM</td>
<td>Lockheed Martin</td>
</tr>
<tr>
<td>ACS</td>
<td>Aegis Combat System</td>
<td>LOT</td>
<td>Launch on TADIL</td>
</tr>
<tr>
<td>ADD</td>
<td>Architecture Definition Document</td>
<td>MMSP</td>
<td>Multi-Mission Signal Processor</td>
</tr>
<tr>
<td>Aegis</td>
<td>(not an acronym) Greek Shield of Zeus</td>
<td>MS</td>
<td>Microsoft</td>
</tr>
<tr>
<td>ALIS</td>
<td>Aegis LAN Interconnect System</td>
<td>NIFC-CA</td>
<td>Naval Integrated Fire Control - Counter Air</td>
</tr>
<tr>
<td>AMOD</td>
<td>Aegis MODernization</td>
<td>OA</td>
<td>Open Architecture</td>
</tr>
<tr>
<td>API</td>
<td>Application Programming Interface</td>
<td>OAET</td>
<td>Open Architecture Enterprise Team</td>
</tr>
<tr>
<td>ASCM</td>
<td>Anti-Ship Cruise Missile</td>
<td>OAET</td>
<td>Open Architecture System Management</td>
</tr>
<tr>
<td>ASROC</td>
<td>Anti-Submarine ROcket</td>
<td>P/S</td>
<td>Publish/Subscribe</td>
</tr>
<tr>
<td>BMD</td>
<td>Ballistic Missile Defense</td>
<td>PIDS</td>
<td>Prime Item Development Specification</td>
</tr>
<tr>
<td>C2</td>
<td>Command and Control</td>
<td>PIM</td>
<td>Platform Independent Model</td>
</tr>
<tr>
<td>CCB</td>
<td>Configuration Control Board</td>
<td>PSEA</td>
<td>Platform System Engineering Agent</td>
</tr>
<tr>
<td>CEC</td>
<td>Cooperative Engagement Capability</td>
<td>PSM</td>
<td>Platform Specific Model</td>
</tr>
<tr>
<td>CG</td>
<td>Guided Missile Cruisers</td>
<td>Pub/Sub</td>
<td>Publish/Subscribe</td>
</tr>
<tr>
<td>CIWS</td>
<td>Close In Weapon System</td>
<td>RF</td>
<td>Radio Frequency</td>
</tr>
<tr>
<td>CM</td>
<td>Configuration Management</td>
<td>SAD</td>
<td>System Architecture Document</td>
</tr>
<tr>
<td>COTS</td>
<td>Commercial Off-the-Shelf</td>
<td>SAN</td>
<td>Storage Area Network</td>
</tr>
<tr>
<td>CPCR</td>
<td>Computer Program Change Request</td>
<td>SBT</td>
<td>Sea-Based Terminal</td>
</tr>
<tr>
<td>CR</td>
<td>COTS Refresh</td>
<td>SI/DA</td>
<td>System Integrator / Design Agent</td>
</tr>
<tr>
<td>CSEA</td>
<td>Combat System Engineering Agent</td>
<td>SM</td>
<td>Standard Missile</td>
</tr>
<tr>
<td>CVN</td>
<td>Carrier Vessel Nuclear</td>
<td>SMP</td>
<td>Symmetric MultiProcessor</td>
</tr>
<tr>
<td>DDG</td>
<td>Guided Missile Destroyer</td>
<td>SQT</td>
<td>System Qualification Test</td>
</tr>
<tr>
<td>DDS</td>
<td>Data Distribution Service</td>
<td>SRS</td>
<td>System Requirements Specification</td>
</tr>
<tr>
<td>DM</td>
<td>Data Model</td>
<td>SSDD</td>
<td>System/Segment Design Document</td>
</tr>
<tr>
<td>DOORS</td>
<td>Dynamic Object-Oriented Requirements System</td>
<td>SSDS</td>
<td>Ship Self Defense System</td>
</tr>
<tr>
<td>ESSM</td>
<td>Evolved Sea Sparrow Missile</td>
<td>SSR</td>
<td>Software Specification Review</td>
</tr>
<tr>
<td>ETR</td>
<td>Engineering Technical Review</td>
<td>STM</td>
<td>System Track Manager</td>
</tr>
<tr>
<td>GCC</td>
<td>GNU Compiler</td>
<td>SVC</td>
<td>Service</td>
</tr>
<tr>
<td>GFE</td>
<td>Government Furnished Equipment</td>
<td>SW</td>
<td>Software</td>
</tr>
<tr>
<td>HM&amp;E</td>
<td>Hull, Mechanical and Electrical</td>
<td>SysML</td>
<td>Systems Modeling Language</td>
</tr>
<tr>
<td>HW</td>
<td>Hardware</td>
<td>T&amp;E</td>
<td>Test and Evaluation</td>
</tr>
<tr>
<td>IAW</td>
<td>In Accordance With</td>
<td>TADIL</td>
<td>TACTical Digital Information Link</td>
</tr>
<tr>
<td>IDD</td>
<td>Interface Definition Document</td>
<td>TI</td>
<td>Technology Insertion</td>
</tr>
<tr>
<td>IDS</td>
<td>Interface Design Specification</td>
<td>TLAM</td>
<td>Tomahawk Land-Attack Missile</td>
</tr>
<tr>
<td>IMS</td>
<td>Integrated Master Schedule</td>
<td>TOR</td>
<td>Test Observation Report</td>
</tr>
<tr>
<td>IPO</td>
<td>Input/Output/Process</td>
<td>TS</td>
<td>Track Server</td>
</tr>
<tr>
<td>IR</td>
<td>Infrared</td>
<td>UML</td>
<td>Unified Modeling Language</td>
</tr>
<tr>
<td>JTM</td>
<td>Joint Track Management</td>
<td>VLA</td>
<td>Vertical Launch ASROC</td>
</tr>
<tr>
<td>KA</td>
<td>Kill Assessment</td>
<td>VLS</td>
<td>Vertical Launch System</td>
</tr>
<tr>
<td></td>
<td></td>
<td>XML</td>
<td>eXtensible Markup Language</td>
</tr>
</tbody>
</table>
Application of Java In AEGIS Weapons Control

Andrew Winkler
Sept 28, 2011
Purpose

- Overview of the use of Java in Aegis Weapons Control Open Architecture (WCOA)
  - Language Selection
  - Early Analysis
- Java Virtual Machine assessment
  - Overview
  - Latest Performance Results
Why Java?

- Faced with a language selection in 2004
  - Development team not trained in C++ or Java
  - Very aggressive schedule
    - Completely re-architect complex ~200 KSLOC shared memory based weapon control program from the top down
    - Complete AAW capability in 36 months.
- Perceived benefits based on initial language assessment (2004)
  - Increased productivity
  - Language features
  - Reduced defects
  - Tools
  - Libraries
- But would Java support performance requirements?
  - Initial Assessment performed in 2004
Early Results (2004)

- VM Run at RT Priority
- Periodic offset by ~10ms
  - Default Sun timer resolution is 10ms
  - Can be set to hi-res – but not used for this measurement
- The one outlier (~7ms) is the first 1st measurement
  - timer resolution?

---

**Sun 8 250ms 80%CPU Load**

<table>
<thead>
<tr>
<th>Time (ms)</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>252</td>
<td>0</td>
</tr>
<tr>
<td>254</td>
<td>0</td>
</tr>
<tr>
<td>256</td>
<td>50</td>
</tr>
<tr>
<td>258</td>
<td>0</td>
</tr>
<tr>
<td>260</td>
<td>10000</td>
</tr>
<tr>
<td>262</td>
<td>0</td>
</tr>
</tbody>
</table>

- Max (ms): 260.214
- Min (ms): 253.306
- Avg (ms): 259.7804
- Std Dev (ms): 0.326611

Copyright © 2011 by Lockheed Martin Corporation
Comparison to C++

Behavior Consistent
With Previous C/C++
Based Measurements
for Sun
Initial Assessment

- Results indicated JVM performance was generally coupled with underlying OS
  - Mainstream JVM on RTOS could exhibit some real time behavior
  - Still issues and watch items
    - Garbage Collection
    - JVM Control (other JVM threads)
    - Threading (priority inheritance)
- Decision was made to proceed with Java
  - Early Data analysis indicated no major obstacles
  - The state of the market
    - Interest among mainstream vendors (BEA, Sun, IBM) for higher performance/deterministic JVMs
    - RT Java support from small vendors (Aicas, Aonix)
    - The emergence of RTSJ
WCOA JVM Technical Requirements

Required

- Ability to map Java thread priorities to underlying OS priorities
- Ability to set Real-time scheduling policy (SCHED_FIFO, SCHED_RR)
- Control over VM threads (e.g. priority, enable/disable)
  - Garbage collection, optimization
- Deterministic behavior
  - Priority inheritance for synchronization
  - Deterministic GC
  - Low Jitter
- Different Compilation options/control
  - Ahead-of-Time compilation
  - Just-in-Time compilation

Desirable

- Support for Real-Time Specification For Java (RTSJ)
Performance Data

- Collected data for several JVMs

- Examined Behavior of GC
  - 50ms periodic thread under load (~50%)
  - CPU load produced by creation and collection of objects
  - Examined instances where periodic ran long
    - Overruns typically caused by GC or another JVM thread such as optimization thread

- Jitter
  - Examine the deterministic behavior of an application over a long period of time (100,000+ data points)
  - Ran 20ms periodic thread under load (~50%)
  - CPU load produced by creation and collection of objects

- Tactical Testing
  - Examined critical timelines running WCOA tactical code
Non-Deterministic Garbage Collection

Trace Events from Periodic thread

50ms Periodic

Periodic Runs Long

Kernel Trace

Copyright © 2011 by Lockheed Martin Corporation
Non-Deterministic Garbage Collection (Continued)

Periodic Runs Long

Periodic Should Have Run Here

GC Thread

Periodic thread runs after GC
Deterministic Garbage Collection

No Overrun Periodics

50ms Periodic

GC Thread
Deterministic Garbage Collection (Continued)

Periodic thread pre-empt GC and runs on time.

Signal to Wake Periodic Thread
Product ‘A’ JVM Jitter Data

<table>
<thead>
<tr>
<th>Average (ms)</th>
<th>Stdev (ms)</th>
<th>Max (ms)</th>
<th>Min (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>28.446</td>
<td>18.150</td>
<td>169.189</td>
<td>20.142</td>
</tr>
</tbody>
</table>
Product ‘B’ JVM Jitter Data

Max Deviation < 1ms: Supports WCOA Requirements
Testing in Tactical Environment

- Performed significant testing using three JVMs using WCOA tactical programs
  - Allowed study of different JVM features in a realistic environment
- Analyzed Impact of JVM threads/features to critical timelines
  - JIT adversely impacted certain critical timelines running under two of the JVMs (one RT and one non-RT)
    - AOT or JIT at initialization solved problem for the RT JVM
    - JIT at init slows application initialization significantly
  - One JVM’s Optimization thread interfered with application health-checking – causing application to be terminated
    - Disabling optimization resolved problem
- GC tuning essential even for Deterministic GC.
  - Poorly tuned RT JVM runs as poorly as NRT counterparts
  - Prevent out of memory conditions paramount

Java Performance Verified In Tactical Testing
What about benefits of using Java?

Did we get the benefits we were hoping for?

- Ada programmers adapted quickly to Java
  - Tools like Eclipse helped transition
  - Developed ~150 KSLOCs in 18 months
  - Portability of Java allowed desktop testing & verification on Windows Platform
    - Verified 3500 Requirements in 5 months
    - 89% 1st time pass rate

Java Performance

- Java can support soft real-time, mission critical applications
- Meets WCOA Performance Requirements

Looking forward

- Keeping an eye on safety critical Java work and it’s implications for a mission critical profile
  - Development must keep in mind value proposition for users

Team Realizing Huge Benefits by Switching to Java