The Modernization of the Aegis Fleet with Open Architecture

Andrew Winkler

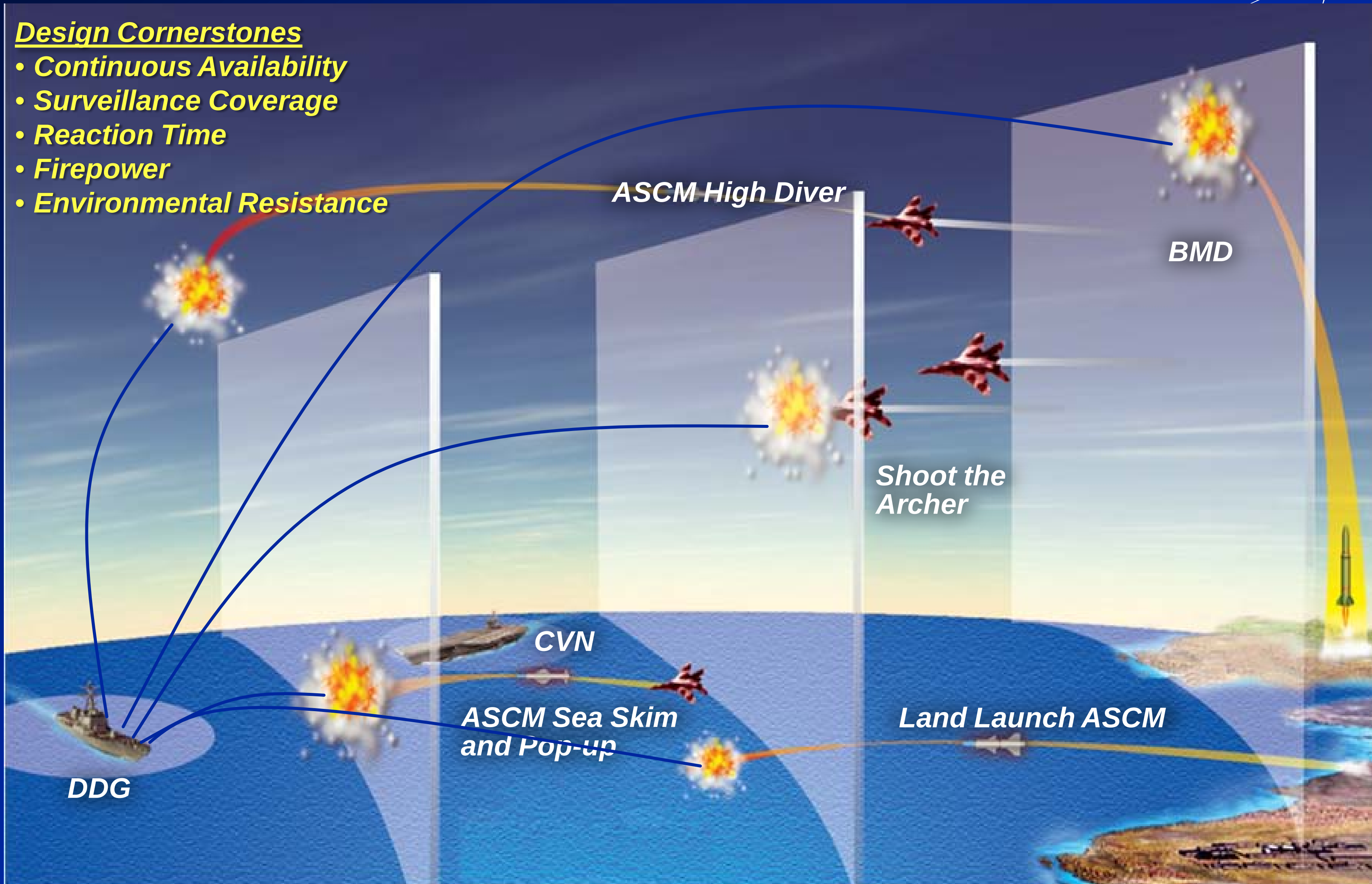
Sept 28, 2011

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**

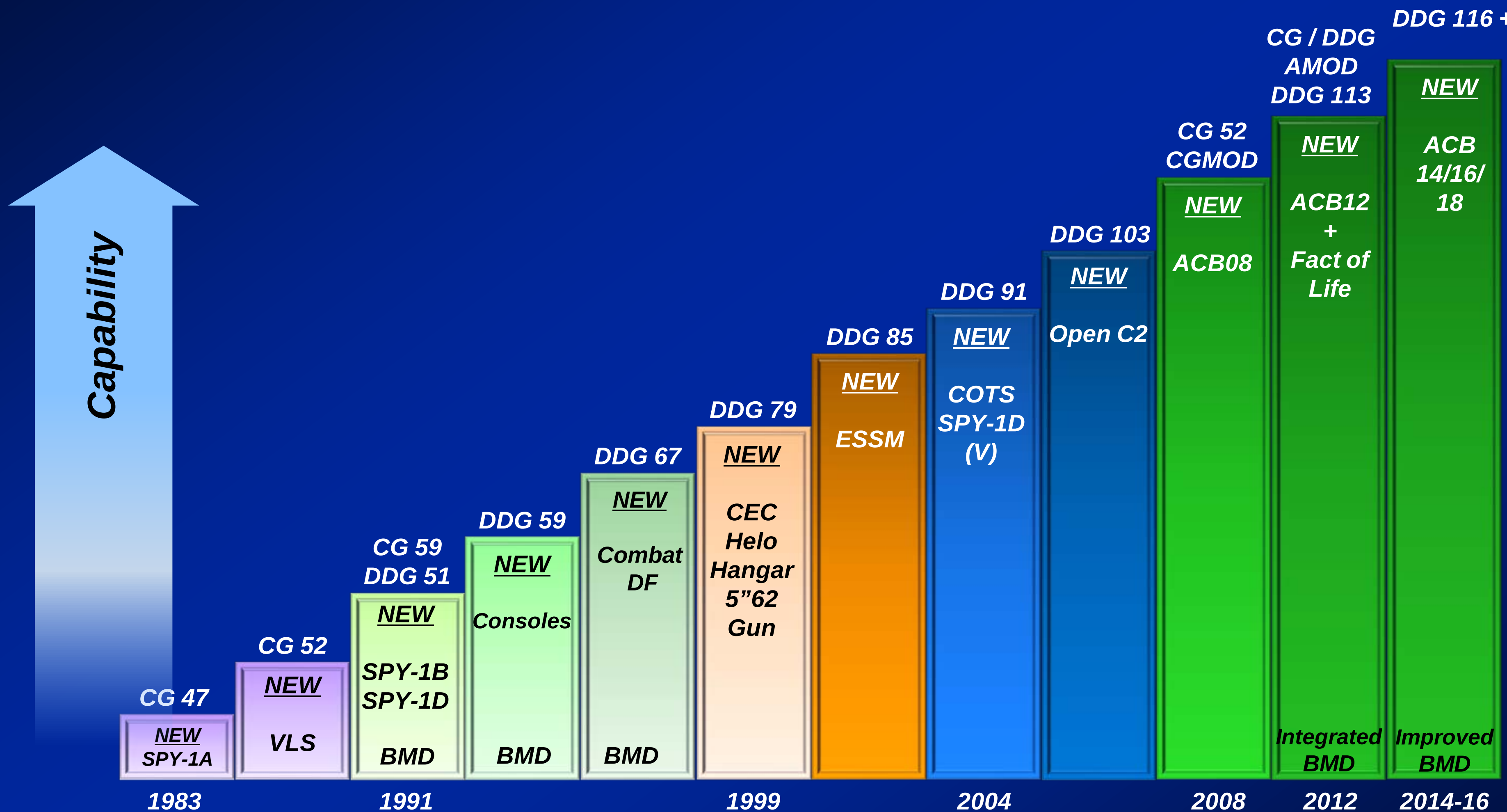
Aegis – The Shield of the Fleet



Self Defense | Area Air Defense | Long Range Air Defense And BMD

Aegis Combat Systems Architecture

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**

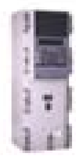
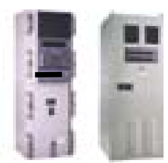
IV. **Transition Aegis to Navy Objective Architecture**

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

	B/L 1/2/3 CG 47-64	B/L 4/5 DDG 51-78 CG 65-73	B/L 6 Ph I/III DDG 79-90 CG 66 & 69	B/L 7 Phase I DDG 91-112	ACB08/TI08 CG 52-58	ACB12/TI12 CG 59+ DDG 51+
Fielding	1983	1990	1998	2005	2009	2012
COTS Selection			1994	1999	2006	2009
						
	MIL Spec Design	MIL Spec Design	Mixed COTS and MIL Spec Design	All COTS computers (MCE)	All COTS computers (MCE)	All COTS computers (MCE, CPS)
Processors	UYK-7 UYK-20 →	UYK-43 UYK-44 →	UYK-43/44+ Adjunct COTS →	COTS →	Mainstream COTS →	Mainstream COTS
Software	CMS-2	CMS-2	CMS-2 C++, Ada	C++, Ada	Message-passing, Component-based C++, Java, Ada	Message-passing, Component-based C++, Java, Ada
Interfaces	NTDS Parallel	NTDS Parallel NTDS Serial	NTDS Parallel NTDS Serial FDDI / Ethernet	NTDS Parallel NTDS Serial FDDI**/ Ethernet	NTDS Parallel NTDS Serial Ethernet	NTDS Parallel NTDS Serial Ethernet
Displays	UYH-4	UYH-4 UYQ-21 (TGC*)	UYQ-21/UYQ-70	UYQ-70	UYQ-70 ORTS Thin Clients	CDS
System Complexities		UYK-43s B5PIII TGC	SMP's ALIS Network	All COTS SMP's Fast Ethernet	Open Architecture Mainstream COTS Gig-Ethernet	Open Architecture COTS MMSP Product-Line

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

COTS Technology and Products

Tech Insertion 00 2000

B7PhI

DDG 91-102

Tech Insertion 04 2004

B7PhIR

DDG 103-112

LCS / NCS - Derivative

Tech Insertion 08 2008

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

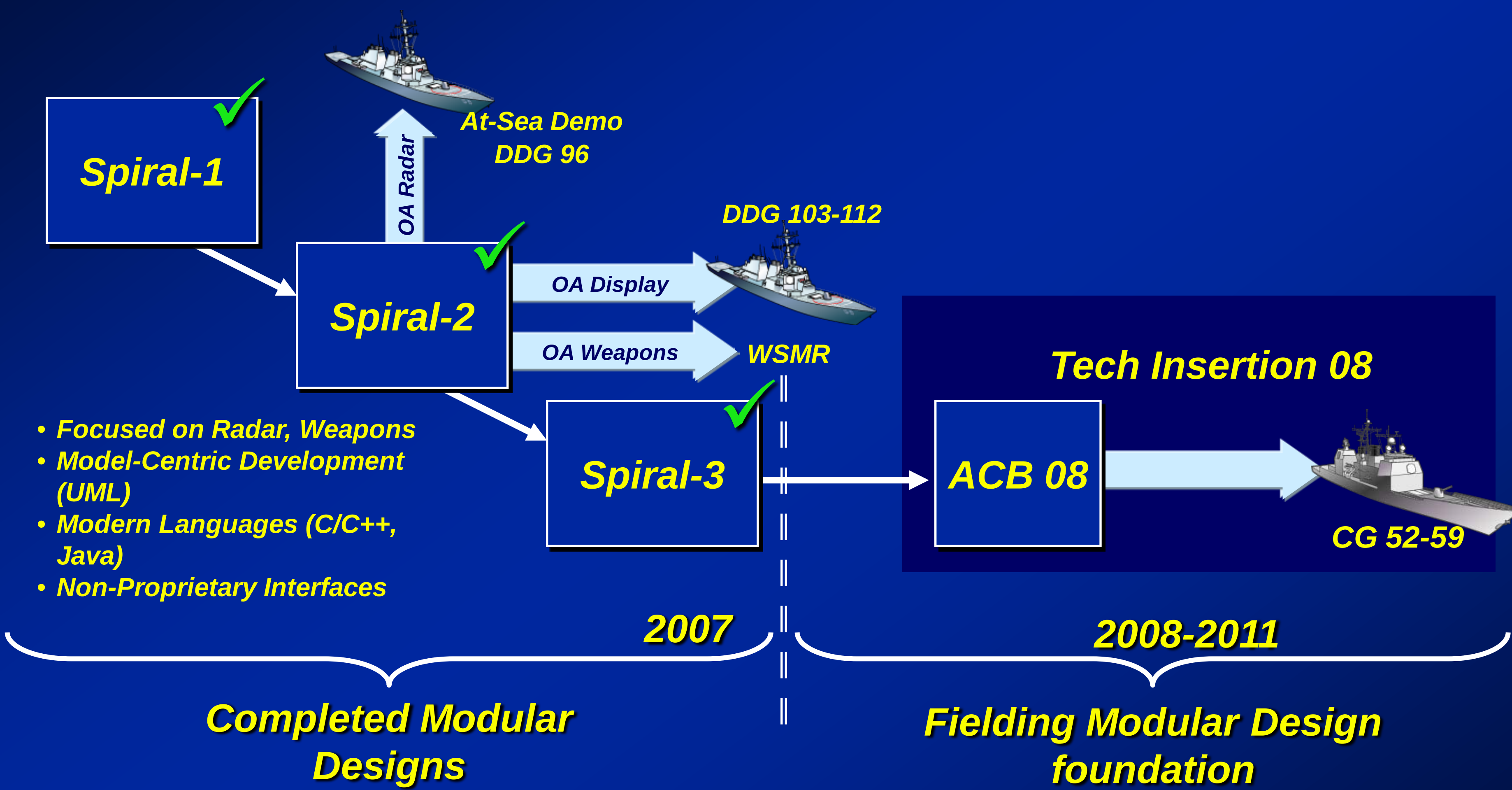
Non-LM Software

- 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

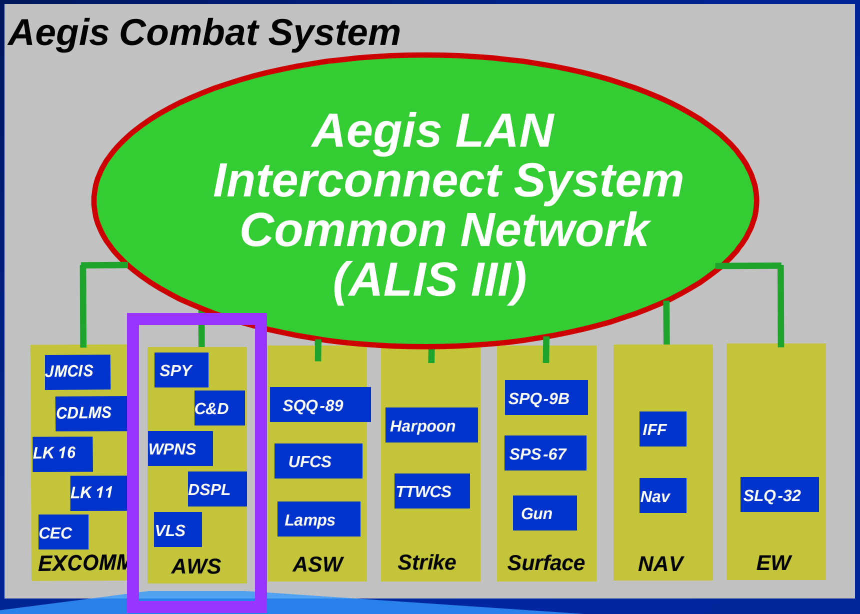
"Build a little ... test a lot"



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

Open Architecture Foundation for Baseline 9 Developments

Where We are Today



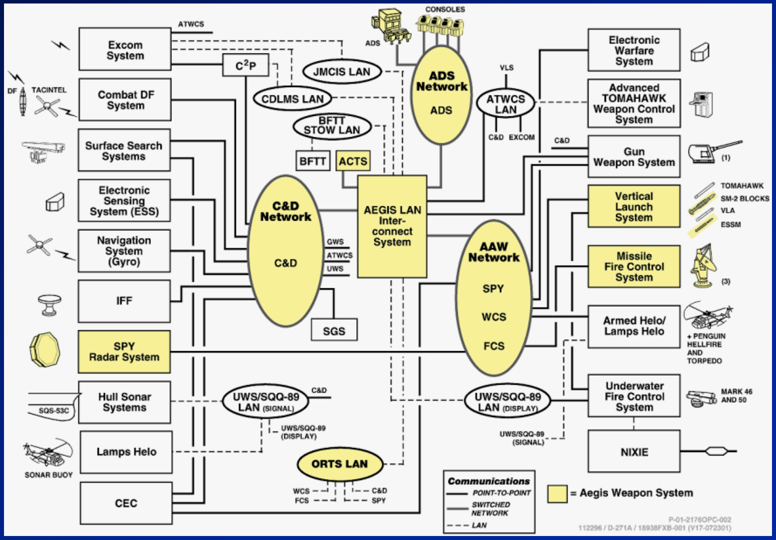
	Technical Assessment	SPY		Open C2		Weapons/ Fire Control	VLS
		Signal Processing	Radar Control	Display	C&D		
DDG-91 2004	Hardware	Custom	SMP	Mainstream	Mainstream	SMP	Mainstream
	Software	Closed	Closed	Open	Open	Closed	Closed
CG-52 2008	Hardware	Custom	Mainstream	Mainstream	Mainstream	Mainstream	Mainstream
	Software	Closed	Open	Open	Open	Open	Closed
CG-62/ DDG-51 2012	Hardware	Mainstream	Mainstream	Mainstream	Mainstream	Mainstream	Mainstream
	Software	Open	Open	Open	Open	Open	Open

Continuously Advancing the Aegis Combat System Forward

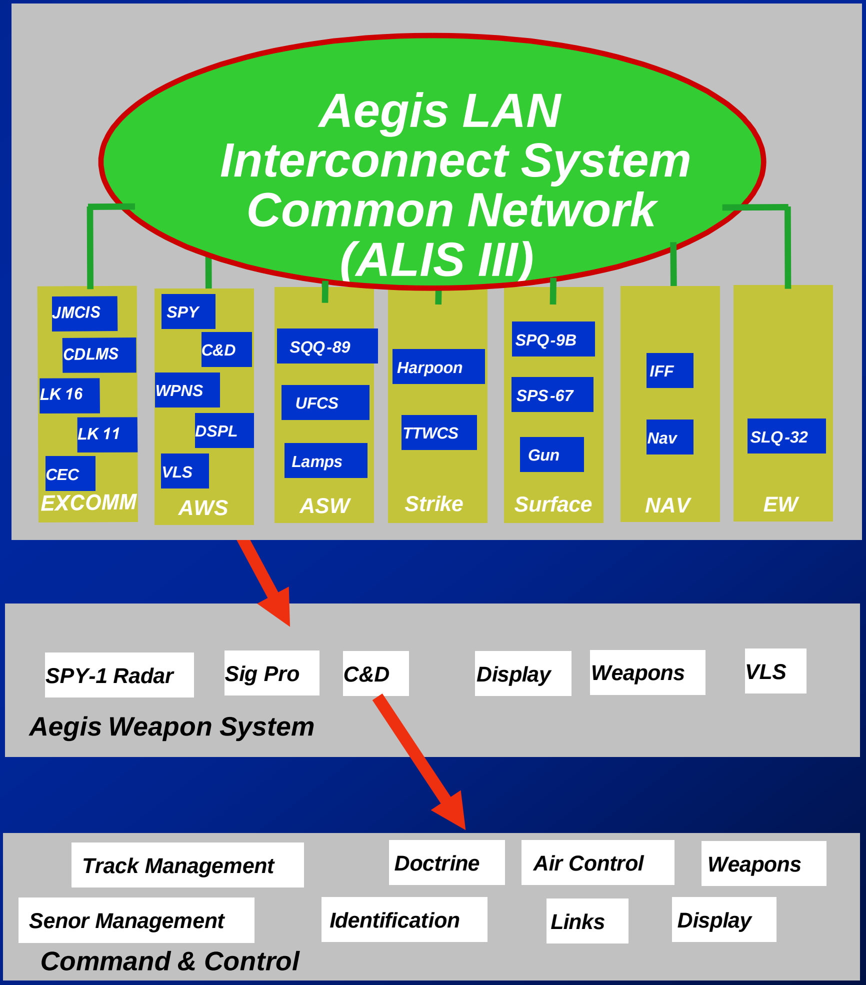
Today's Aegis Combat System

Surface Warfighting Electronics Architecture

Detect/Control/Engage View



System/Subsystem View

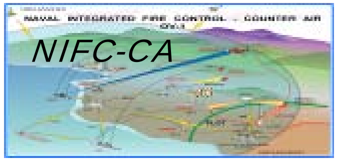


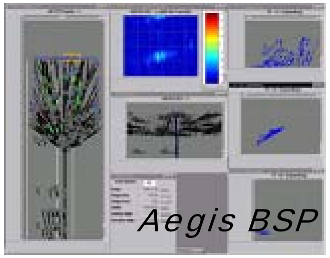
- ✓ ***Federated, Tiered Architecture***
- ✓ ***Efficient ACS Capability changes***
- ✓ ***Well-Define Components and API's***

Supports Operational and Navy Business Model Objectives

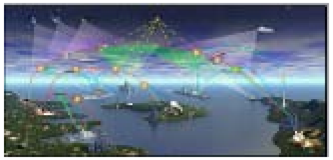
Aegis Modernization

Roadmap to Aegis Modernization (AMOD)

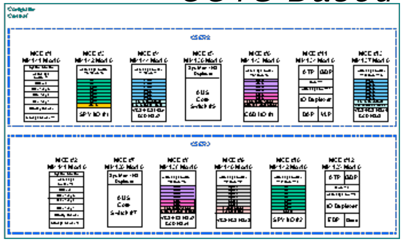




Increased Battlespace and Multi-Mission Interoperability



COTS Based Infrastructure



AMOD Advanced Capability Build 12 (DDG configuration)

- **Tech Insertion (TI) 12**
- **ACS Element Upgrades**
- **JTM Alignment**
- **SM-6**
- **Aegis BMD 5.0**
- **NIFC-CA**
- **MMSP**

AMOD Advanced Capability Build 12 (CG configuration)

- **TI12**
- **ACS Element Upgrades**
- **JTM Alignment**
- **SM-6**
- **NIFC-CA**

AMOD ACB12 (TI12)

Aegis BMD 4.0.1

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

ACB08

- **OA Spiral 3**
- **ACS Element Upgrades**
- **TI 08**

CG MOD ACB08 (TI08)

Aegis BMD Block 06/08

Aegis BMD 3.6

- **LRS&T, Engagement and LoT**
- **Multi-Mission**
- **Integrated Mission Planning**
- **SM-3 Blk I and IA**

B/L 7 Phase IR

- **OA Display Improvements**
- **CIWS Block 1B Fratricide Avoidance**
- **CEC 2.1 (Mode 5)**
- **COTS Refresh 1**

B/L 7 Phase I

- **COTS architecture**

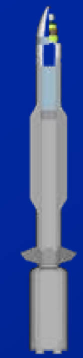
Aegis BMD Block 04

CR0/CR1

AMOD Technical Scope

HM&E Upgrades

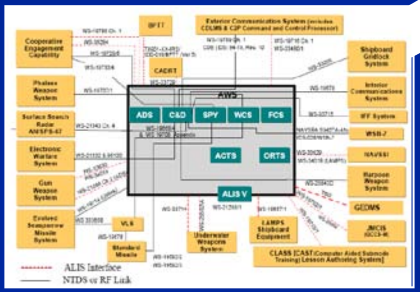
BMD 4.0.1 Functionality and SM-3

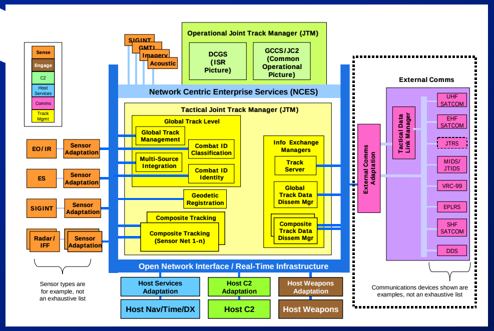


Aegis Weapon System COTS Refresh 3

NIFC-CA and SM-6

Aegis Combat System Upgrades





JTM Alignment

Way Ahead … Baseline 9

Spiral-1

Spiral-2

Spiral-3

Focus: SPY, Weapons, and Display

Steering and Priming
The SBIR Process

"ACB 08"

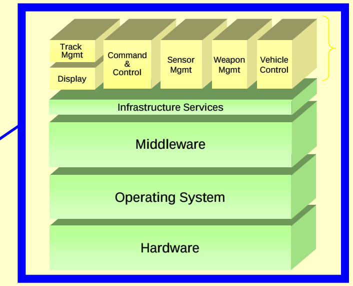
USS Bunker Hill CG-52

- *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

Technology Collaboration Center

Integrate 3rd Party



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

ACB 8.1	Baseline 9 A/C	Baseline 9 B/D/E..
	MM Signal Processor	MM Signal Processor
• Capability Improvements	• BMD Merger • NIFC-CA • JTM / ATAC	• SBT • NIFC-CA Update
		
2010	2012	2014-2016

Field Capability Through Modernization

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

Implementing Open Architecture

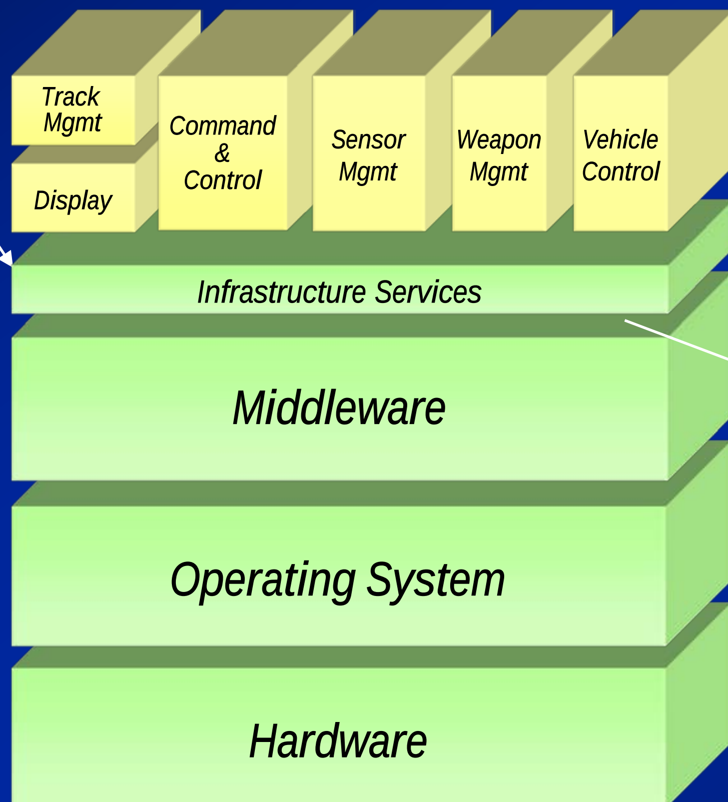
Layered Architecture Foundation

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:

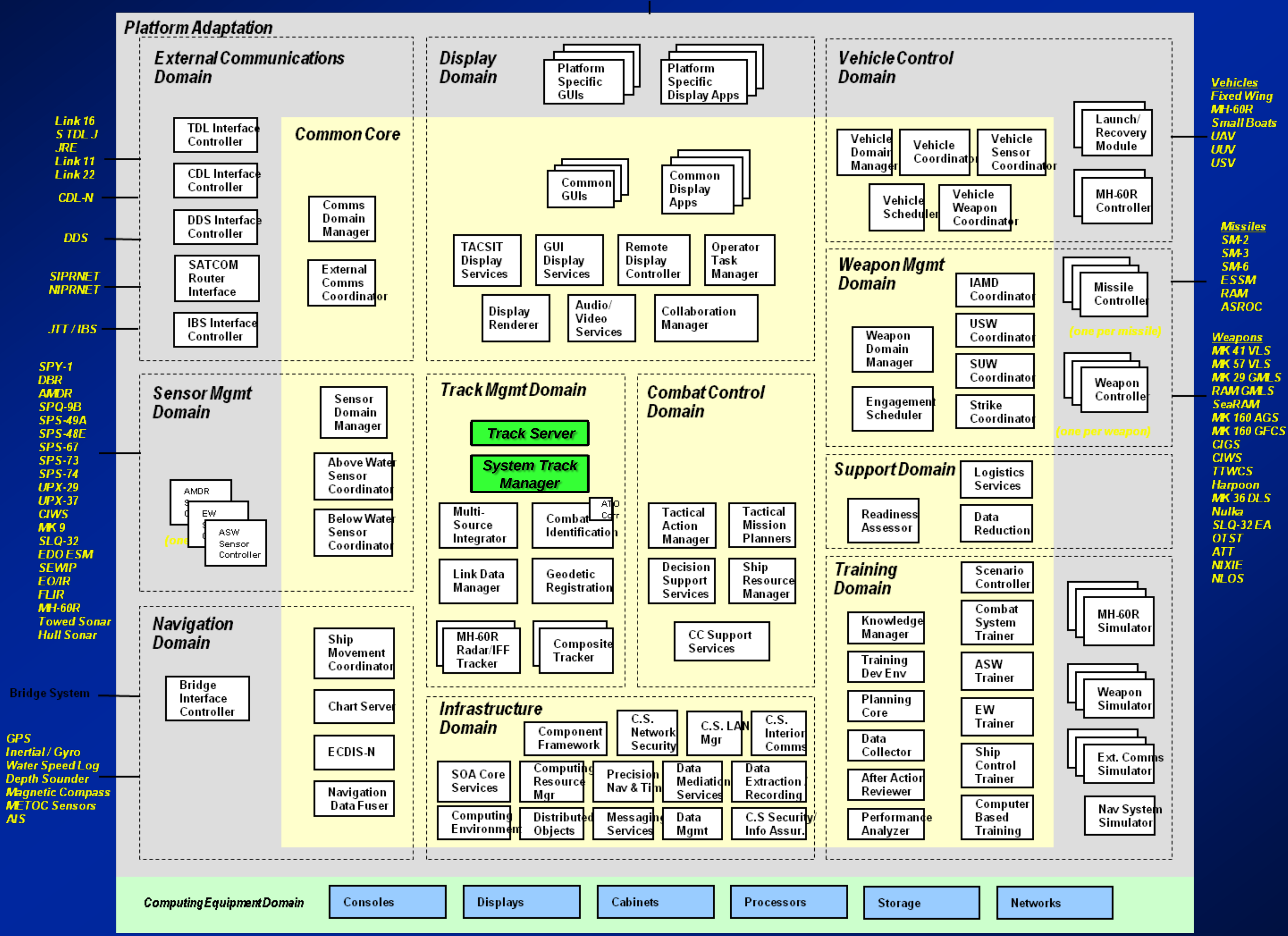
- Common Reusable Components
- Platform Specific Components
- Data Model
- Extensible to the Future

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

Upgrading Hardware and Software Independently

Top Level Objective Architecture

"Component View…"



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 <u>System Track Manager</u> and <u>Track Server</u> 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 Ownship – Tactical and Training:**
  - Allows Training View to be Repositioned with No Impact to Tactical View

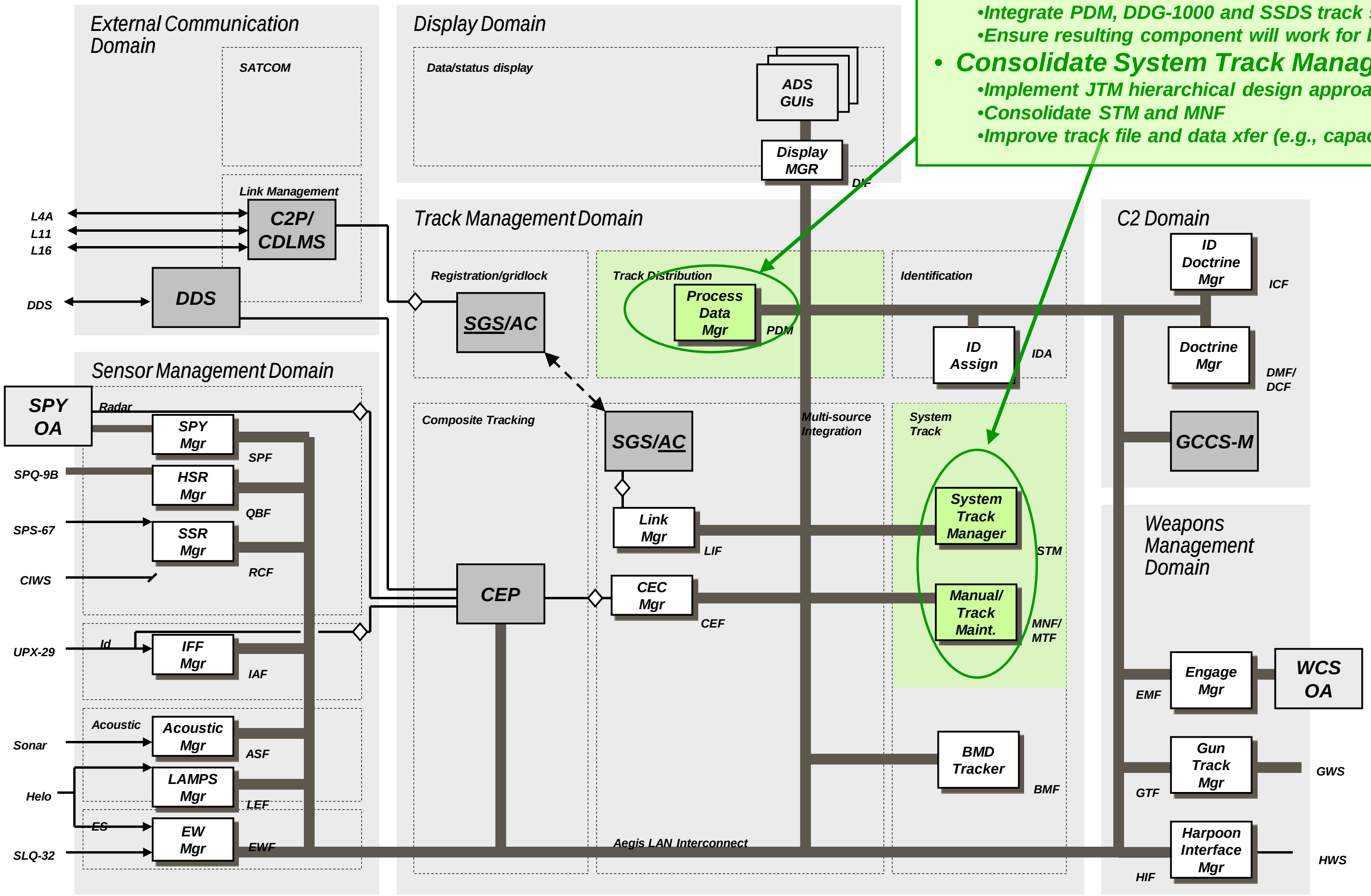
Aligning the Architecture for Future:
<u>Common Components</u> Across Ship Classes

JTM Alignment

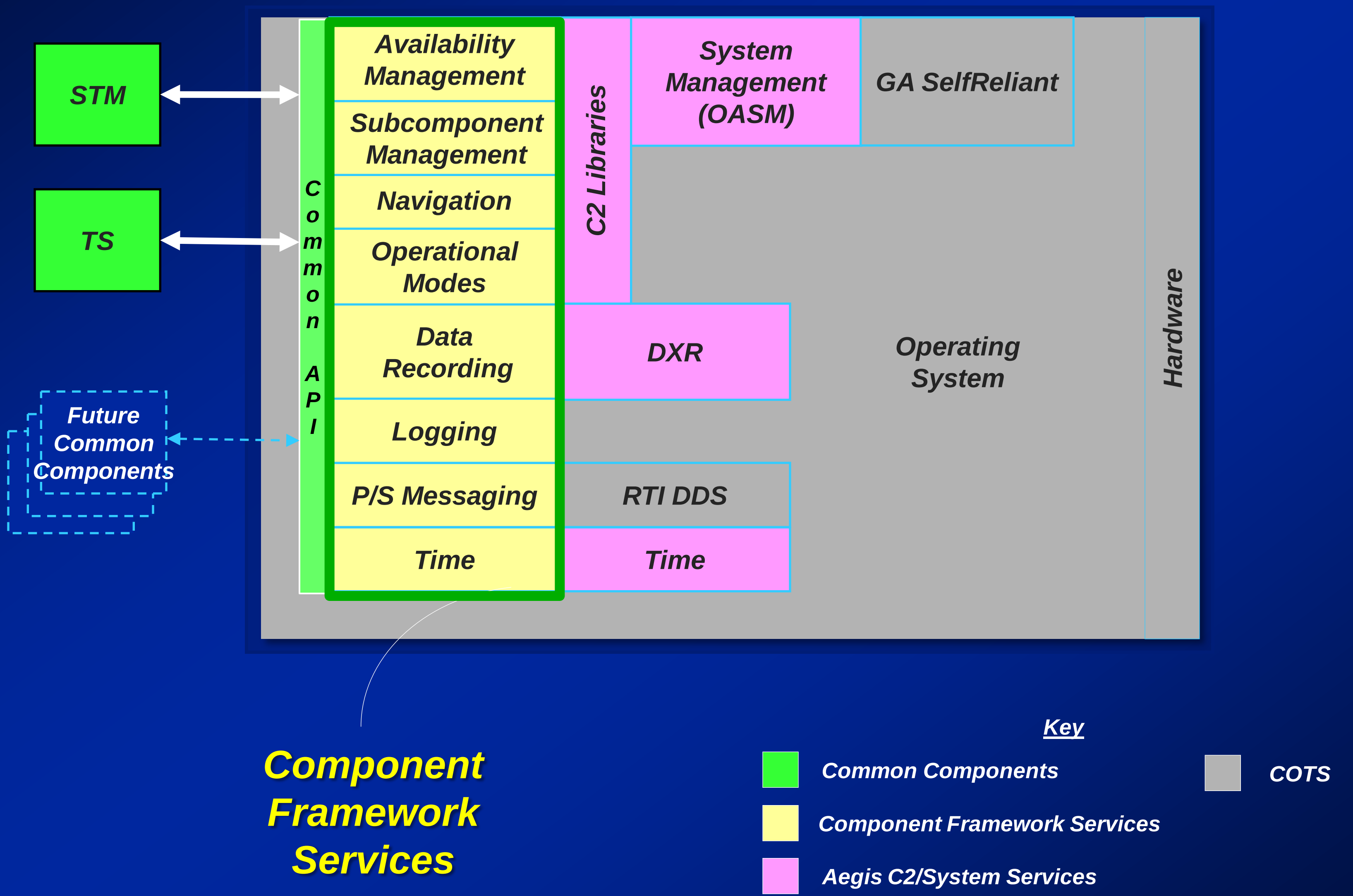
Integration of Common STM and TS Components…

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

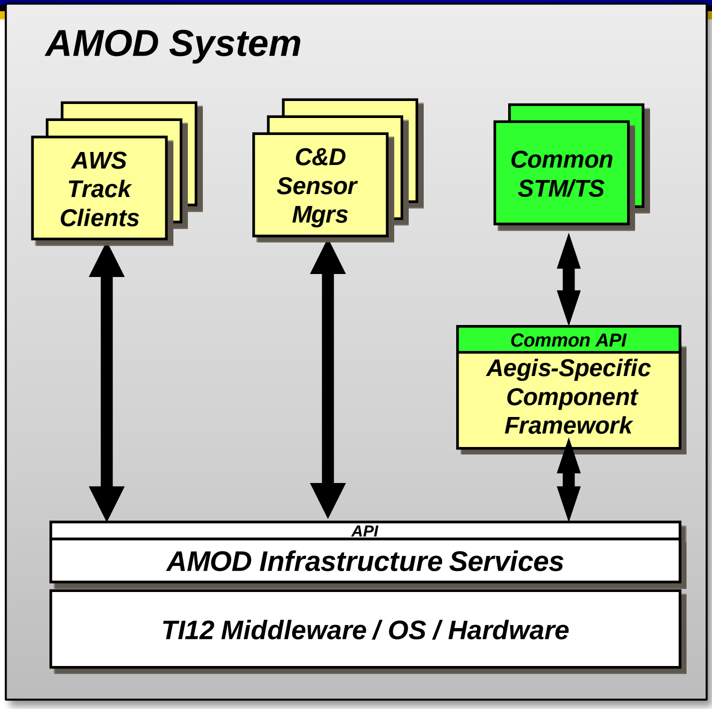


Common STM and TS Components

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:

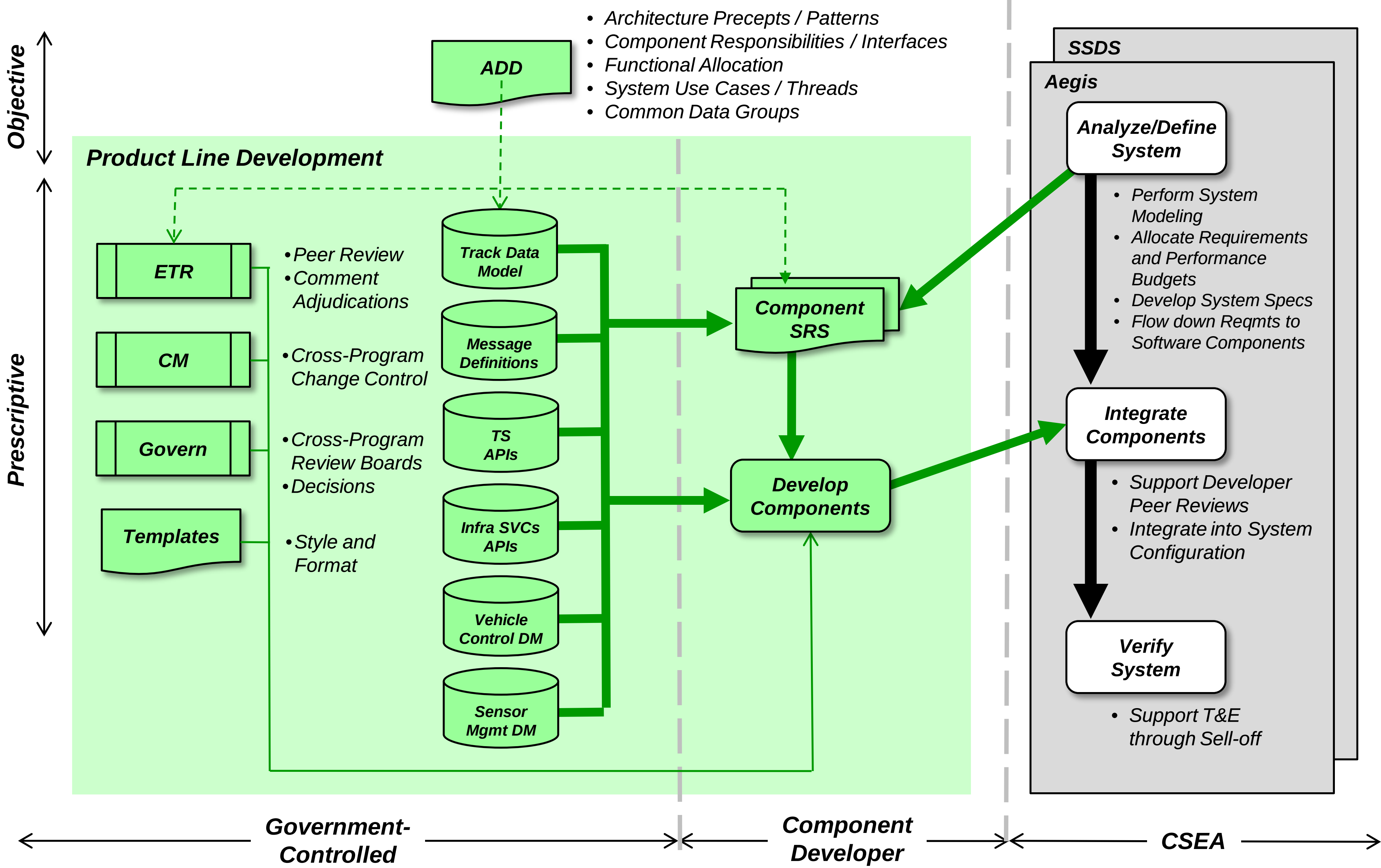
- *Establish linked classified development environment*
- *Establish and Track Progress and Dependencies via Joint IMS*
- *Participate in Navy-led Data Model and Component Framework Working Groups*
- *Support Functional Allocation*
- *Support Definition of Data Model, TS APIs, and Common Service APIs*
- *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 (yellow)
- New Common (green)

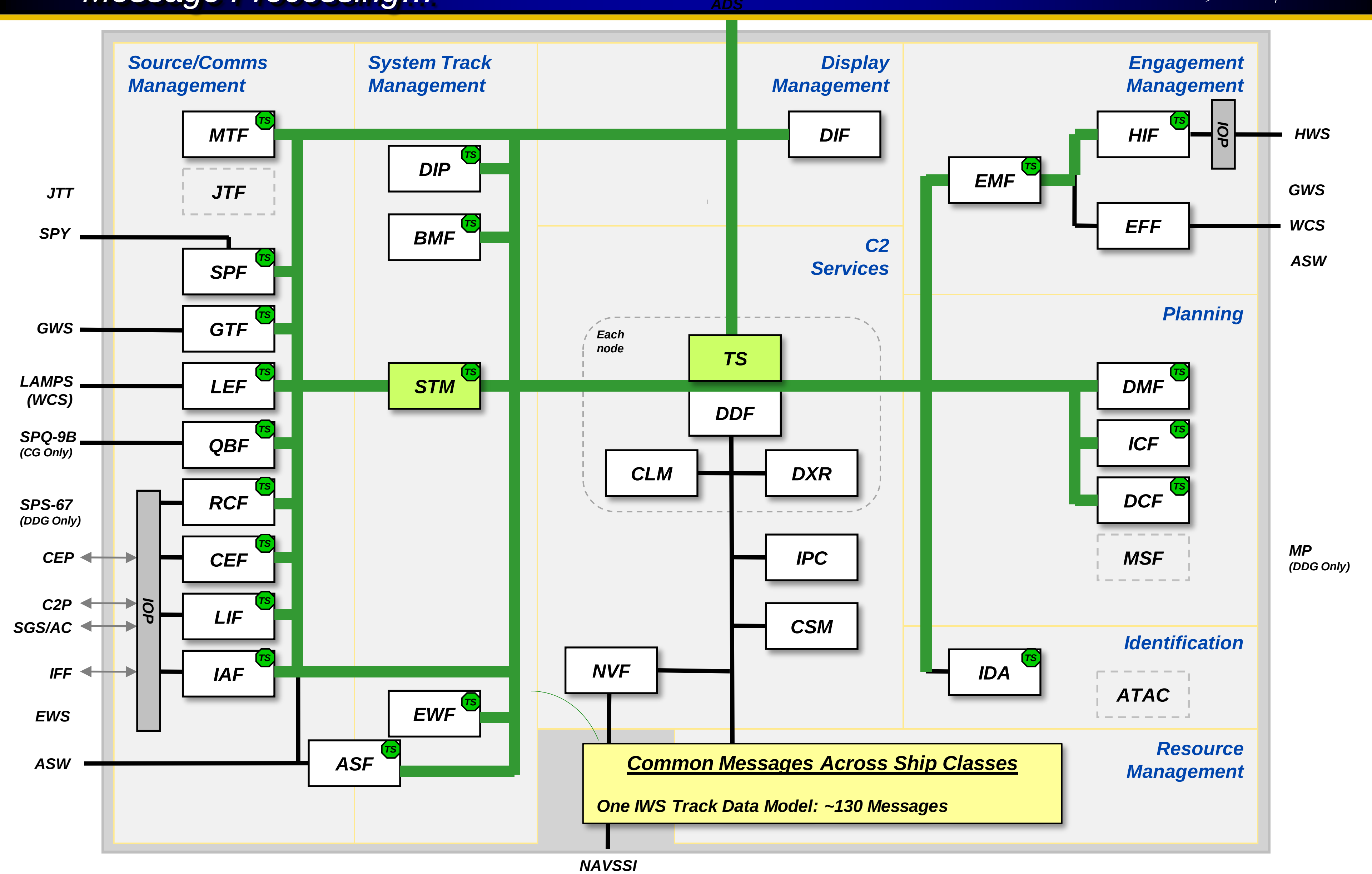
Allocation and Governance Was Essential

Objective Architecture
Roles and Responsibilities…



AMOD C&D Component Architecture

Message Processing…



What We Learned

Process

- ***Documentation***
- ***Design/Integration Tools***
- ***Software Development***
- ***Test Environment***

People

- ***Skills / Expertise***
- ***Organization***

Technical

- ***Architecture***
- ***Functional***
- ***Performance***

Programmatic

- ***Dependencies***
- ***Risks***
- ***Earned Value***

Lessons Learned Address <u>Multiple Perspectives</u>

Aegis Open Architecture

Summary

1994 — 2000 — 2006 — 2012-2016

COTS Infrastructure

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

B7PhI

DDG-91+

Component-Based Software

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

CGM

CG-52+

Open Business/ Common Components

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

AMOD

DDG-51+ / DDG 113

CG-62-73

Increased Capabilities

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

Glossary

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

Application of Java In AEGIS Weapons Control

Andrew Winkler

Sept 28, 2011

Purpose

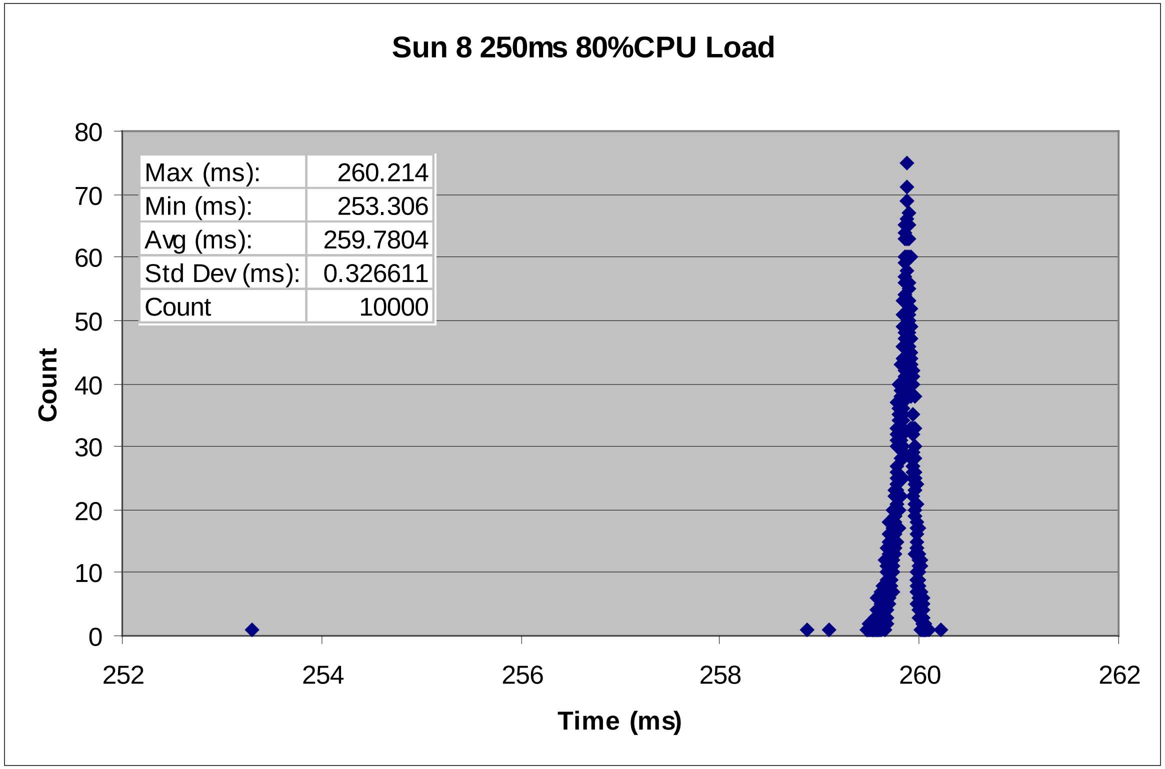
- Overview of the use of Java in Aegis Weapons Control Open Architecture (WCOA)
  - Language Selection
  - Early Analysis
- Java Virtual Machine assessment
  - Overview
  - Latest Performance Results

Why Java?

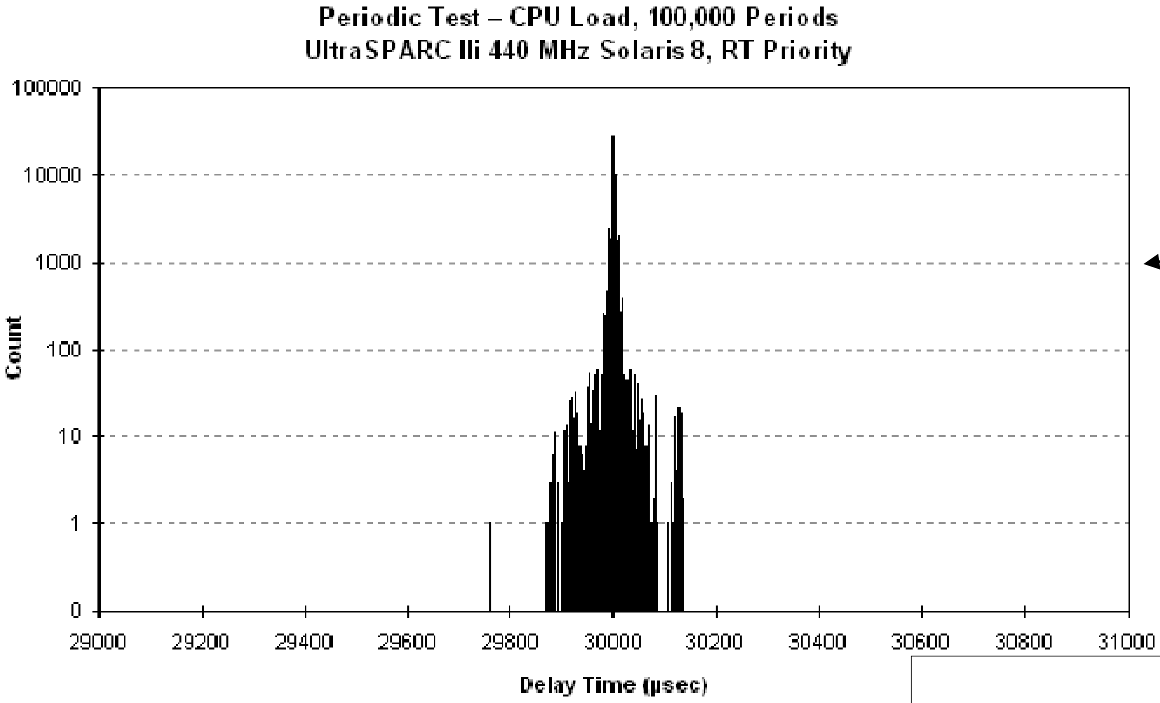
- Faced with a language selection in 2004
  - Development team not trained in C++ or Java
  - Very aggressive schedule
    - Completely re-architect complex ~200 KSLOC shared memory based weapon control program from the top down
    - Complete AAW capability in 36 months.
- Perceived benefits based on 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?

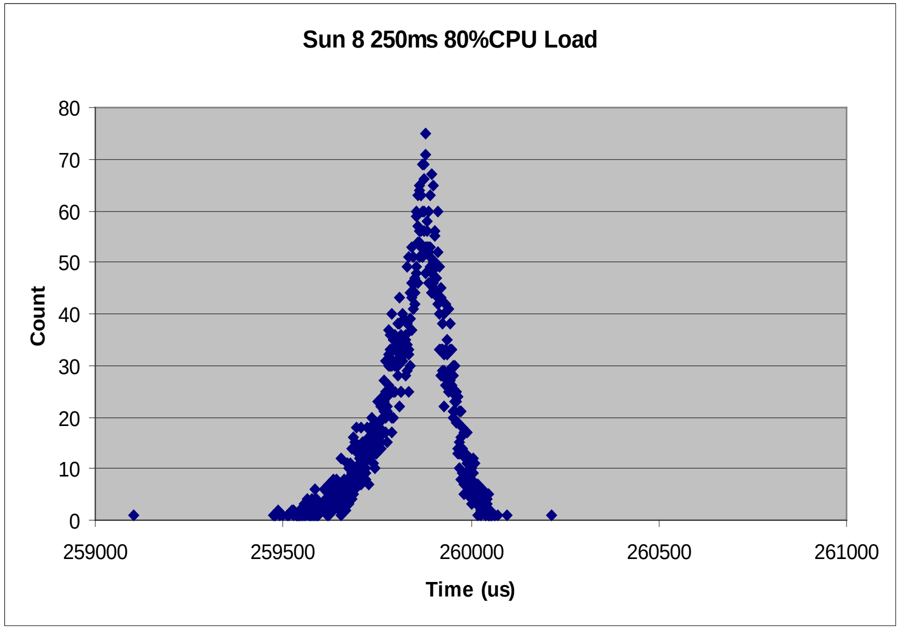


Comparison to C++



Charts on same 2ms scale

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

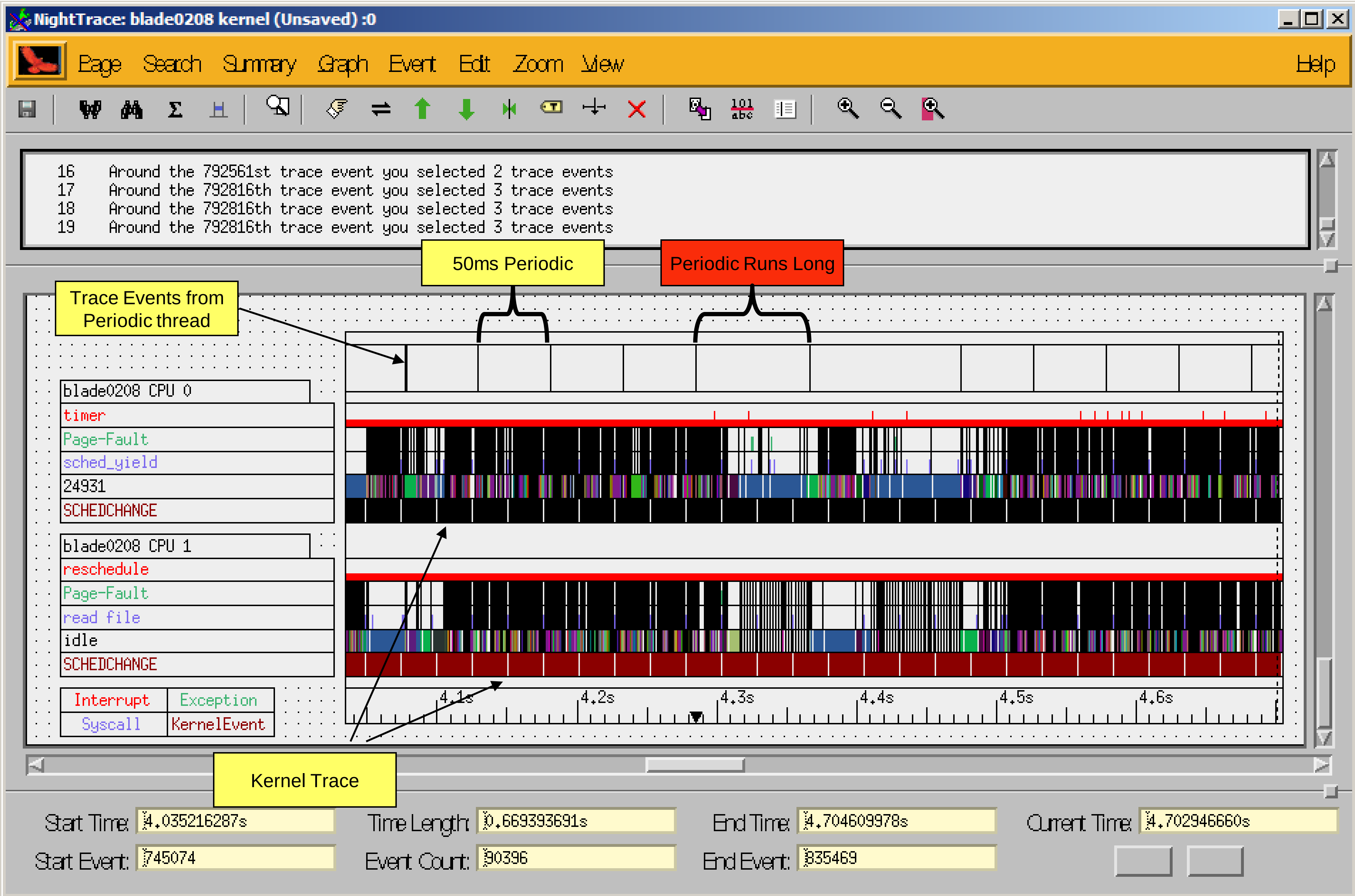
WCOA JVM Technical Requirements

- **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)

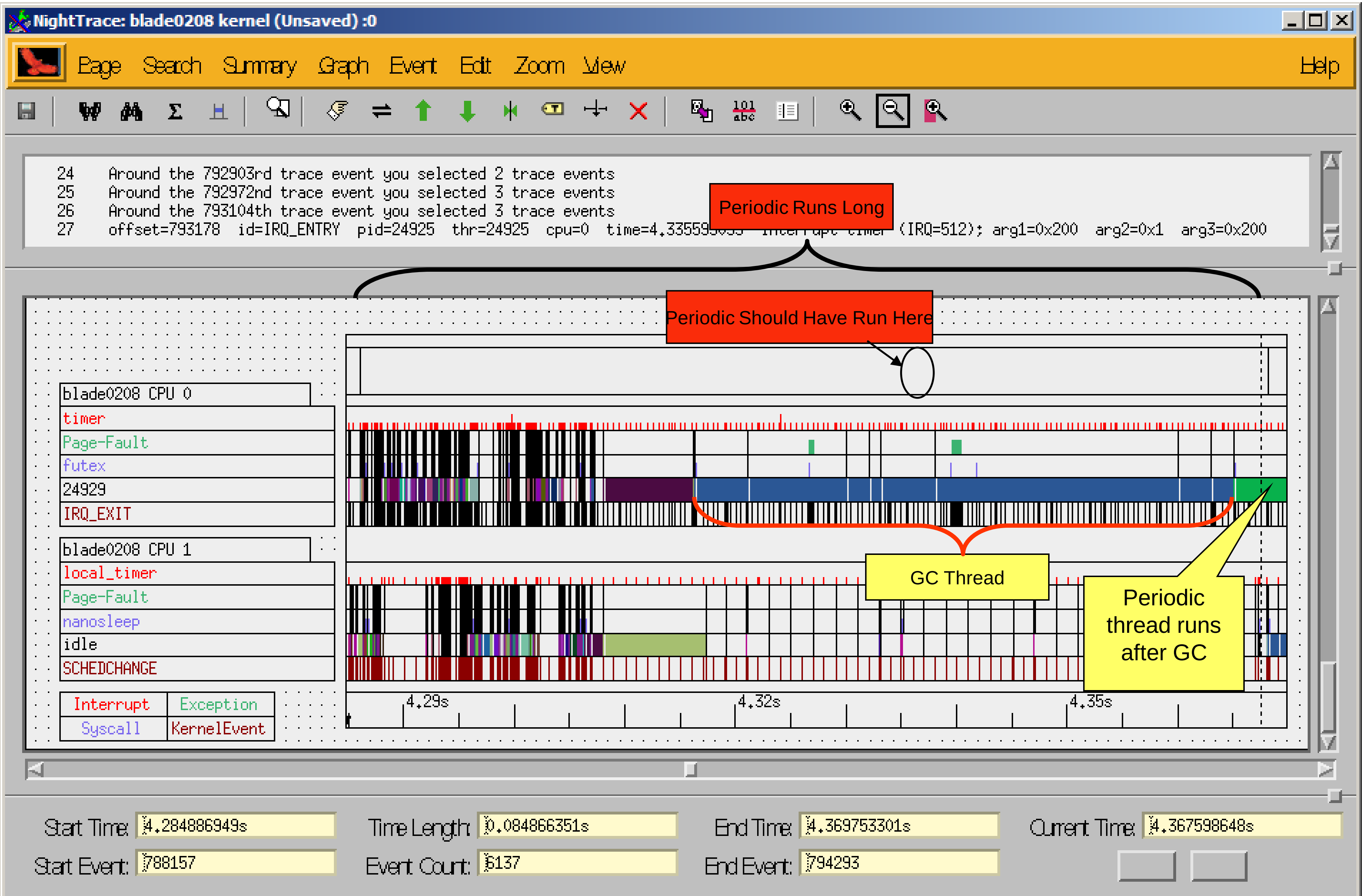
Performance Data

- 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

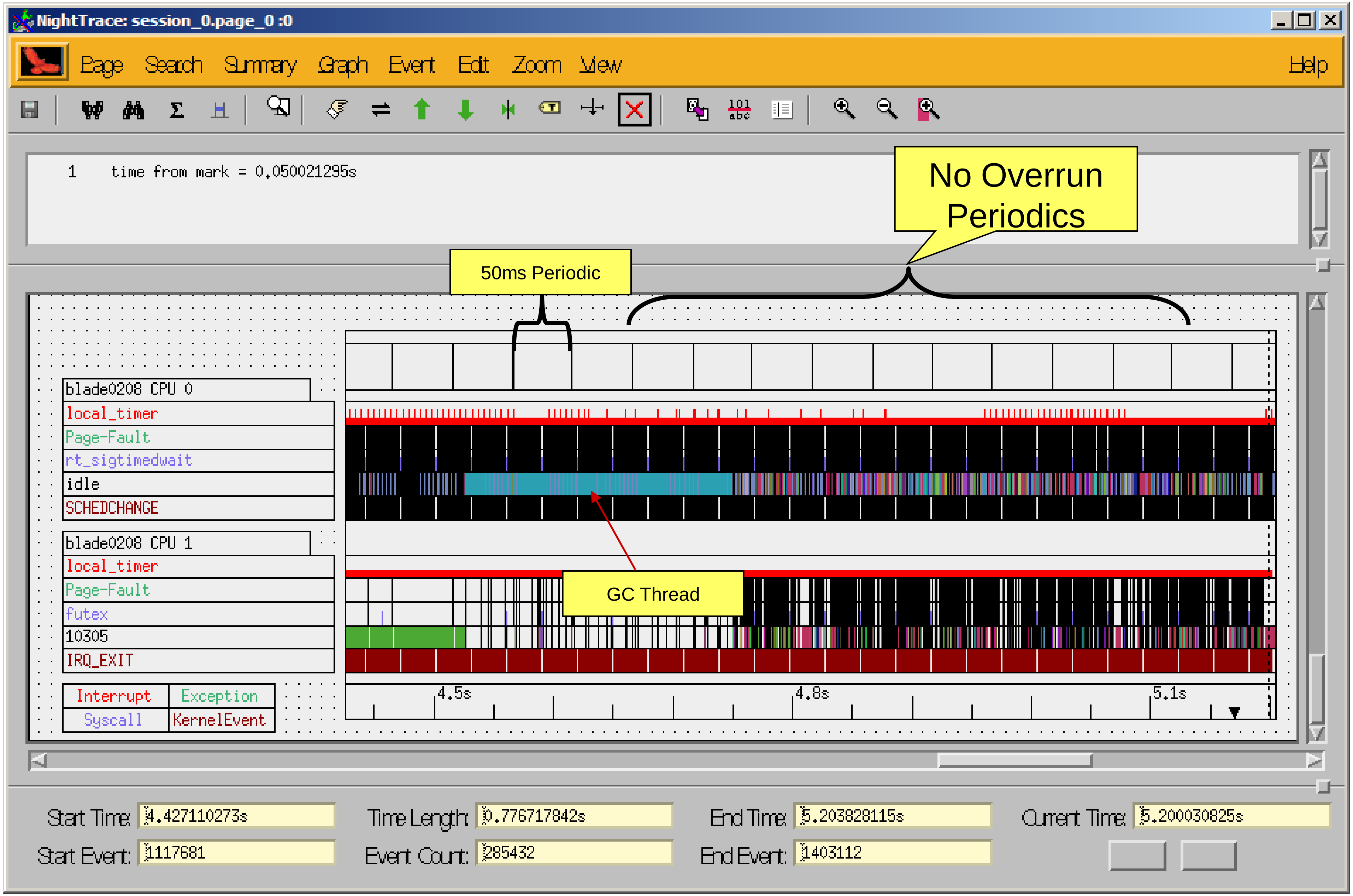
Non-Deterministic Garbage Collection



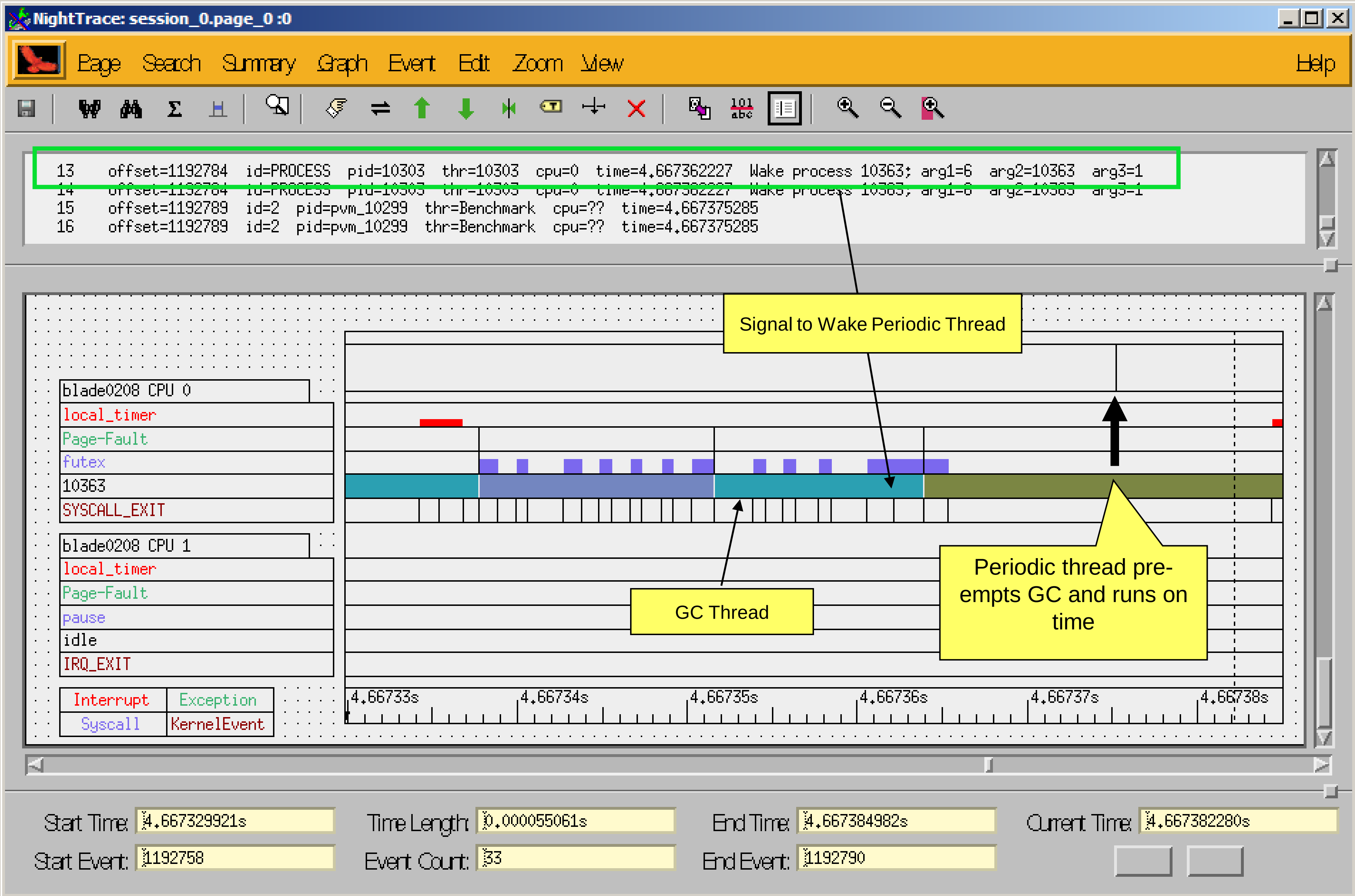
Non-Deterministic Garbage Collection (Continued)



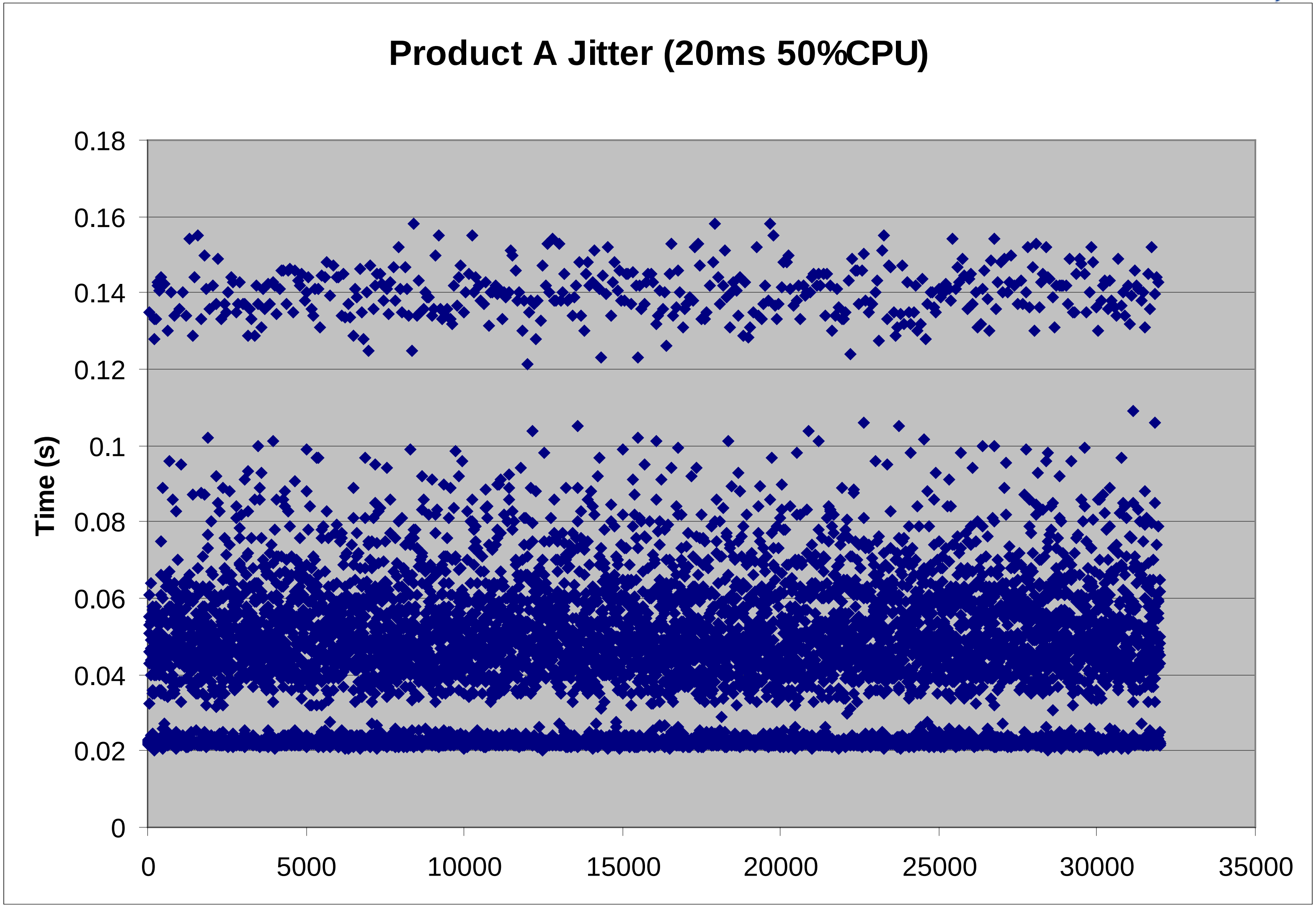
Deterministic Garbage Collection



Deterministic Garbage Collection (Continued)

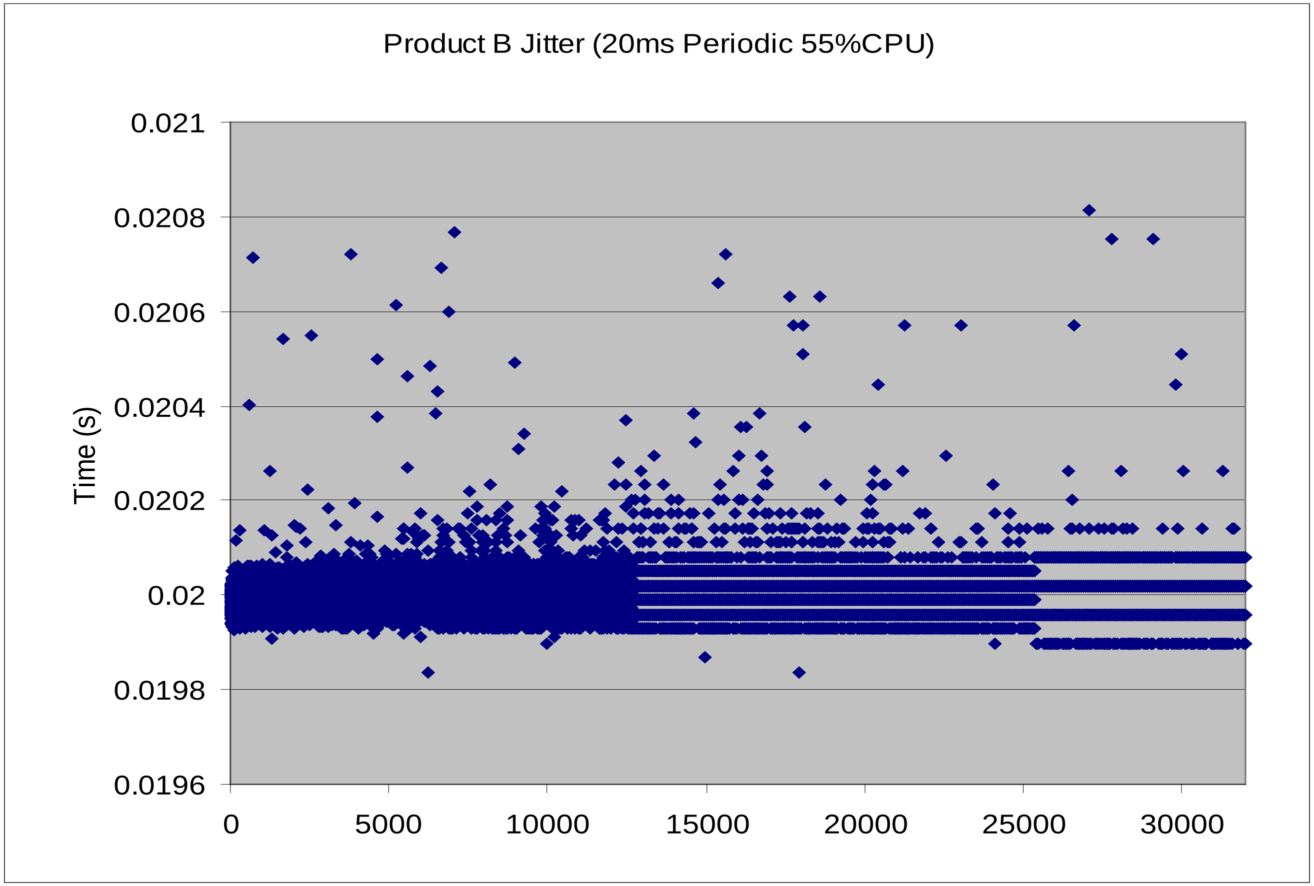


Product ‘A’ JVM Jitter Data



Average (ms)	Stdev (ms)	Max (ms)	Min (ms)
28.446	18.150	169.189	20.142

Product ‘B’ JVM Jitter Data



Average (ms)	Stdev (ms)	Max (ms)	Min (ms)
19.997	0.056	20.935	19.775

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