MSE

MISSION SOLUTIONS ENGINEERING

Performance Engineering Initiatives for Early Software Test of High Availability Systems

Ed Beck
Sr. Manager

NDIA Test & Evaluation Conference
March 1-4, 2010

Mission Solutions Engineering, LLC

- Mission Solutions Engineering (MSE) is a **full service systems and software engineering provider** with 40 years' experience in delivering mission systems.

- Formed by CSC to address government concerns for potential OCI, MSE comprises CSC's former systems and software engineering support to the US Navy and Missile Defense Agency, primarily operating out of Moorestown, New Jersey and with headquarters in Arlington, Virginia.

- The core of our engineering activities is the design, development and integration of mission-critical software. We have been rated at Capability Maturity Model Integration1 Level 5 for software engineering (SW-CMMI®) since 2001 and systems engineering (SE-CMMI®) since 2004.

Aegis Weapon System

U.S. Navy Photo

- The world’s most advanced shipboard Anti-Air Warfare (AAW) Weapons System
- A highly integrated Combat System capable of Multi-Mission warfare
  - Air, Surface and Subsurface
- Open Architecture migration provides foundation for the modernization effort and future war fighting capabilities

Overview

- Performance Engineering (PE), Integration & Test
- Catalyst for PE Initiatives
- Performance Engineering Framework
- Summary

U.S. Navy Photo

What is Performance Engineering?

- **Performance Engineering is an emerging Computer Science practice comprised of the following functions:**
  - Capacity Planning
    - Projection of future resource needs based on historical data and growth projections
  - Performance Measurements
    - Demonstrating that the system meets performance criteria
  - Performance Analysis
    - Characterizing software behavior and identifying anomalies

- **In practice, as we have defined it at MSE, Performance Engineering is the process by which software is tested and tuned with the intent of realizing the required performance.**
  - Viewed at MSE as an integral function within the Test & Evaluation (T&E) Discipline

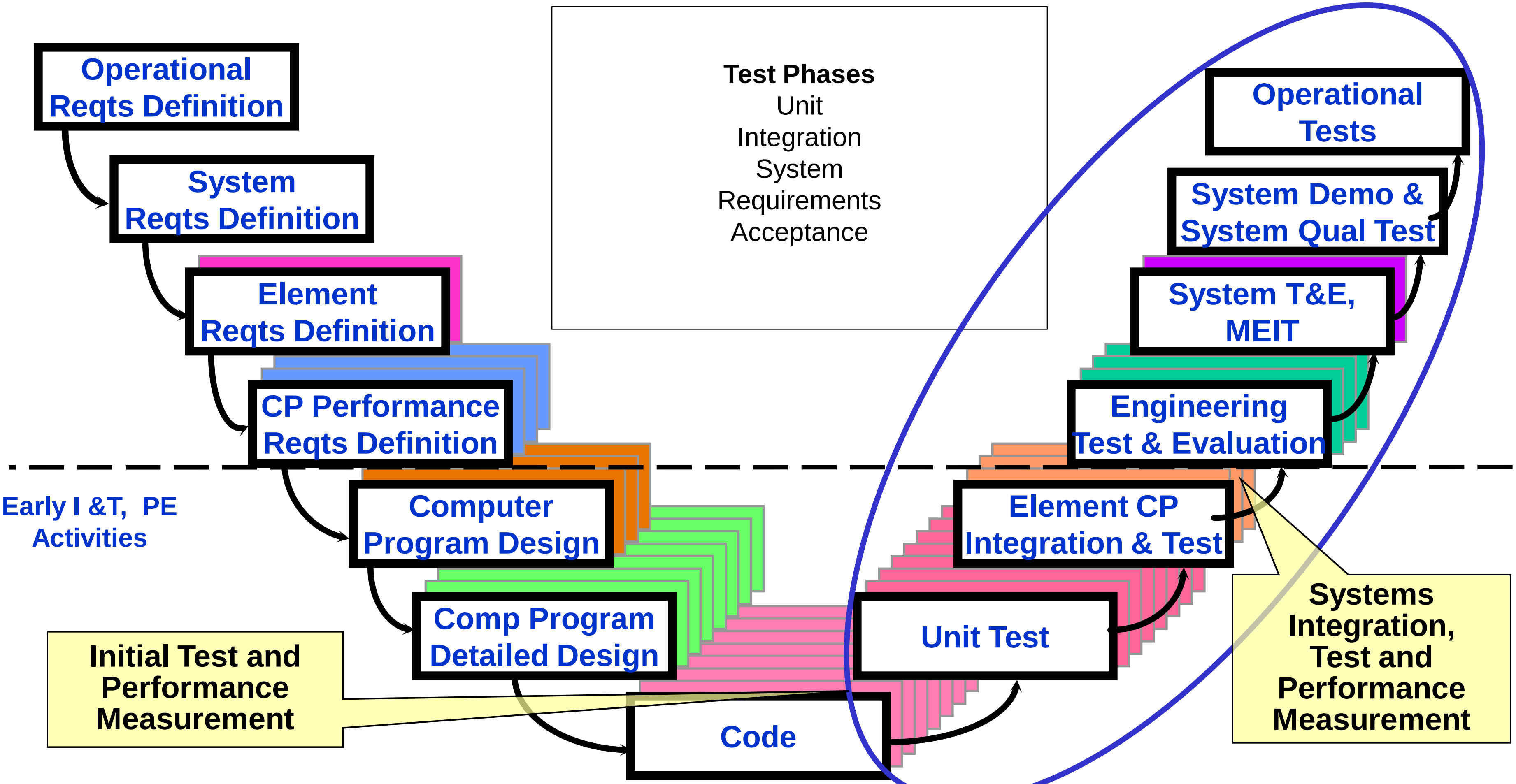
What is Systems Integration & Test?

- **Systems Integration & Test is the ability to verify and validate the interfaces, functions and performance of two or more computer programs on the target hardware suite and operating environment.**

- **Multiple Integration & Test levels:**
  - Unit
    - Verifies functionality of specific sections of code
  - Integration
    - Verifies the interfaces and interactions of components against a specific design
  - System
    - Verifies the system performs intended functions and integrates with external or 3rd party interfaces
  - Requirements
    - Verifies that a completely integrated systems meets requirements
  - Acceptance
    - Validates the system meets customer needs

Multiple Testing Levels

System Definition & Design	Computer Program Definition & Design	Computer Program Implementation	Computer Program Testing	System Integration Testing



Test Phases
Unit
Integration
System
Requirements
Acceptance

Operational Reqts Definition

System Reqts Definition

Element Reqts Definition

CP Performance Reqts Definition

Computer Program Design

Comp Program Detailed Design

Code

Unit Test

Element CP Integration & Test

Engineering Test & Evaluation

System T&E, MEIT

System Demo & System Qual Test

Operational Tests

Early I &T, PE Activities

Initial Test and Performance Measurement

Systems Integration, Test and Performance Measurement

Early Integration, Test & Performance Engineering

- **Software testing efforts account for a large part of the software development costs**
  - The earlier a defect is found, the cheaper it is to fix

- **Incorporating Performance Engineering initiatives in Early Integration & Test efforts results in:**
  - Better characterization of software and system performance levels
  - Earlier detection of performance defects which yield lower remediation costs
  - ***Timely integration of 3rd party software solutions with better operational performance***

Catalyst for Performance Engineering Initiatives

U.S. Navy Photos

A Migration to Open Architecture

Proprietary Systems

Manufactured hardware and developed software

U.S. Navy Photo

Open Technology

Emphasis on COTS hardware and software integration

C/C++

Java

Conversion of legacy software to new languages

concurrent

Sun microsystems

redhat

hp invent

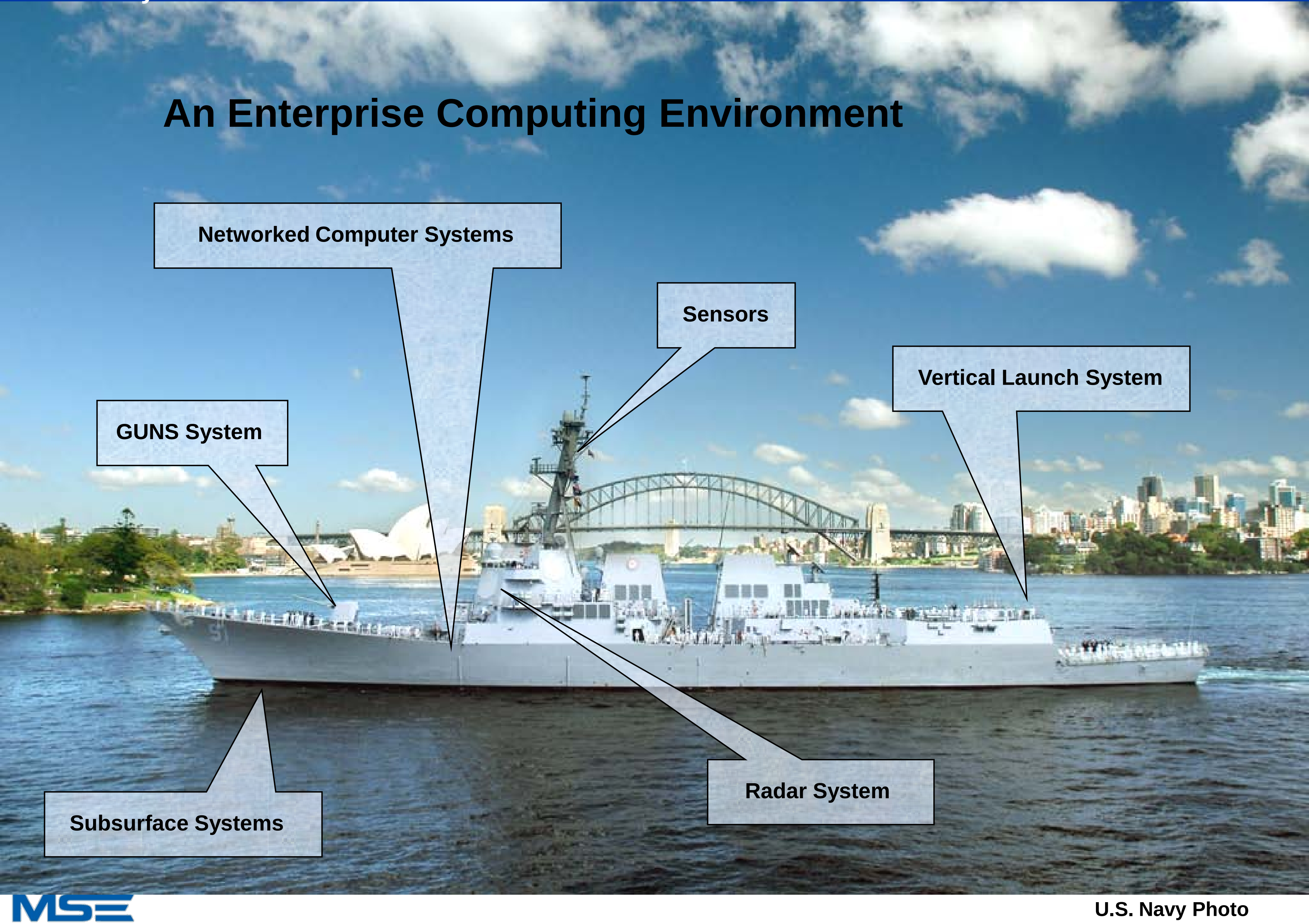
Heterogeneous operating environments

Open Technology

- An Aegis ship is not much different from a large-scale commercial data center.

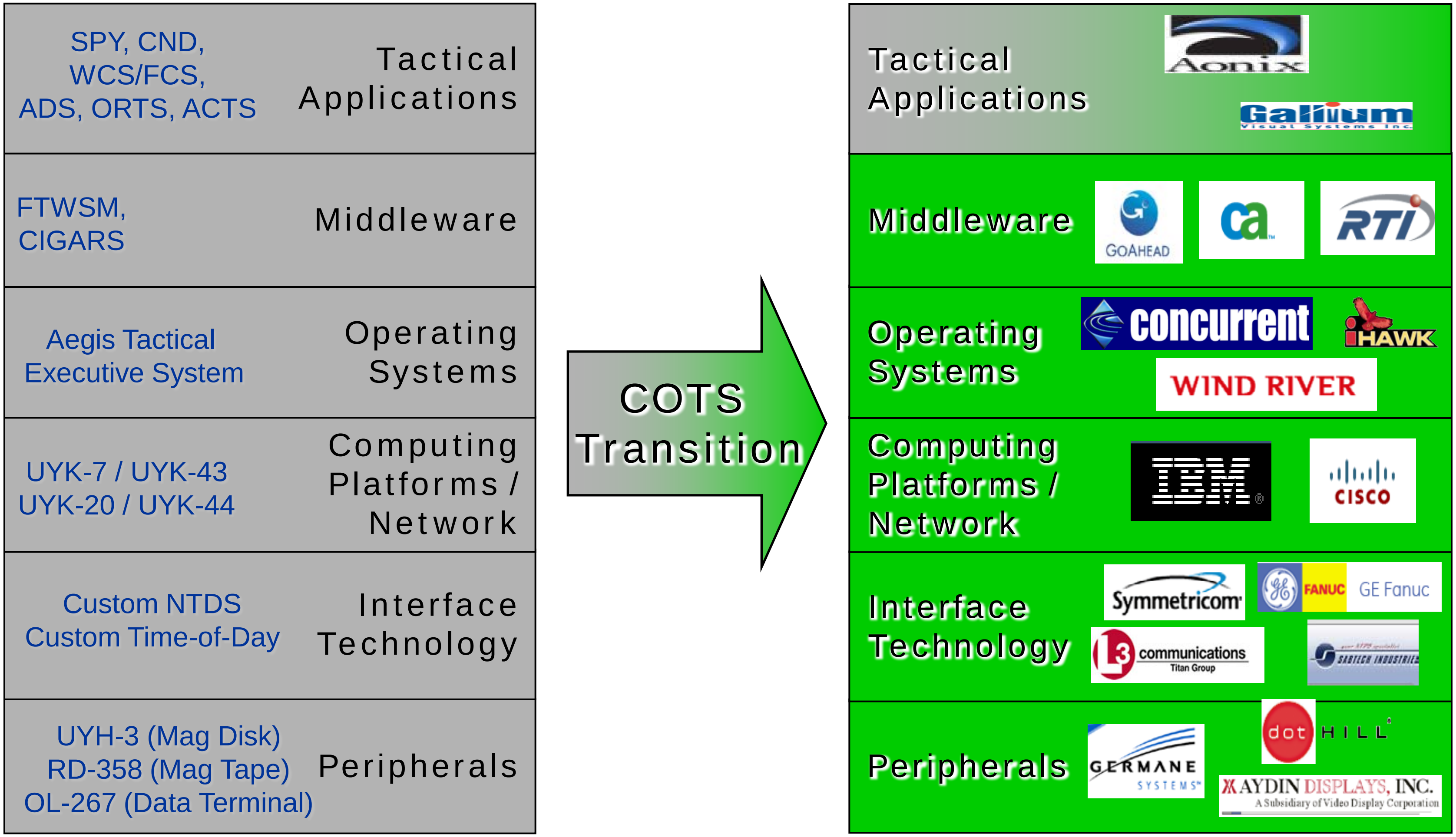
- The weapons system is composed of a heterogeneous operating environment with unique components not seen in commercial architectures.

An Enterprise Computing Environment



U.S. Navy Photo

Transformation to Open Architecture – Numerous challenges with integrating COTS technology for Real-Time High Availability computing requirements



SPY, CND, WCS/FCS, ADS, ORTS, ACTS	Tactical Applications	Tactical Applications	Aonix Gallium Visual Systems Inc.
FTWSM, CIGARS	Middleware	Middleware	GOAHEAD ca RTI
Aegis Tactical Executive System	Operating Systems	Operating Systems	concurrent iHAWK WIND RIVER
UYK-7 / UYK-43 UYK-20 / UYK-44	Computing Platforms / Network	Computing Platforms / Network	IBM cisco
Custom NTDS Custom Time-of-Day	Interface Technology	Interface Technology	Symmetricom GE FANUC GE Fanuc L3 communications Titan Group SBS Technologies / Sabtech Industries
UYH-3 (Mag Disk) RD-358 (Mag Tape) OL-267 (Data Terminal)	Peripherals	Peripherals	GERMANE SYSTEMS dot HILL AYDIN DISPLAYS, INC. A Subsidiary of Video Display Corporation

Performance Engineering Framework

U.S. Navy Photo

Performance Engineering Initiatives for Aegis Modernization

- **Analysis of Software Vulnerabilities**
  - Static SW scans and run-time monitoring
  - Cppcheck tool (Open Source Software) for SW scans
- **Infrastructure Performance Measurements**
  - Cpu ,Memory, Network Interface utilization
  - Collectd tool (Open Source Software)
- **System/Software Profiling**
  - Process, function and code level graphical & textual reports
  - Oprofile tool (Open Source Software)
- **System-Level Technical Performance Measures (TPMs)**
  - Derived from A-spec requirements
  - Data Extraction and Reduction
- **Product-Area TPMs**
  - Derived from the B5 spec or assumed requirements
  - Data Extraction and Reduction
- **Predictive Engineering**
  - Analysis of planned effects of modifications to the system
  - SW performance based on capacity changes

*** Refer to appendix for tool origin**

Extensive Use Of Open Source Technology

- **Off the shelf system management and performance analysis products are commercially available, but they are:**
  - Targeted for commercial IT applications
  - Not easily adaptable to address specific user requirements
  - Highly complex solutions that require significant technical skills and training
  - Not cost effective: high licensing fees, support and training costs

- **Open Source Benefits:**
  - DoD acceptance of OSS
    - Oct 16, 2009 DoD Memorandum for Secretaries of the Military Departments: Clarifying Guidance Regarding Open Source Software (OSS)
  - Lower cost ‘generic’ alternative to COTS products
  - OSS available at all levels of the software stack
  - Scalable, extensible components provide for innovative solutions

Toolset

Tool	Description	Network Analysis	CPU Utilization	Memory Utilization	Memory Leak Detection	Software Profiling
cppcheck	Performs static analysis of C/C++ source code for software vulnerabilities				X	
valgrind	Performs run-time analysis of executables for instances of memory leaks				X	
collectd	Provides dynamic display of system and application performance – cpu, memory and network utilization. Historical data is available for trend analysis.	X	X	X	X	
ntop	Provides run-time data on network utilization	X				
NightTune	Provides dynamic display of system and application performance – cpu, memory and network utilization. Historical data can be correlated with NightTrace data-set for TPM analysis		X	X	X	
NightTrace	Provides synchronized graphical or text display of all system activity					X
Data Recording	Provides analysis of tactical run-time extraction point data					X
oprofile	System-wide profiler for Linux systems, capable of profiling all running code at low overhead					X
wireshark	Provides run-time analysis of detailed network packet information	X				

* Refer to appendix for tool origin

Focus of Today's Presentation

Tool	Description	Network Analysis	CPU Utilization	Memory Utilization	Memory Leak Detection	Software Profiling
cppcheck	Performs static analysis of C/C++ source code for software vulnerabilities				X	
valgrind	Performs run-time analysis of executables for instances of memory leaks				X	
collectd	Provides dynamic display of system and application performance – cpu, memory and network utilization. Historical data is available for trend analysis.	X	X	X	X	
ntop	Provides run-time data on network utilization	X				
NightTune	Provides dynamic display of system and application performance – cpu, memory and network utilization. Historical data can be correlated with NightTrace data-set for TPM analysis		X	X	X	
NightTrace	Provides synchronized graphical or text display of all system activity					X
Data Recording	Provides analysis of tactical run-time extraction point data					X
oprofile	System-wide profiler for Linux systems, capable of profiling all running code at low overhead					X
wireshark	Provides run-time analysis of detailed network packet information	X				

* Refer to appendix for tool origin

PE Framework

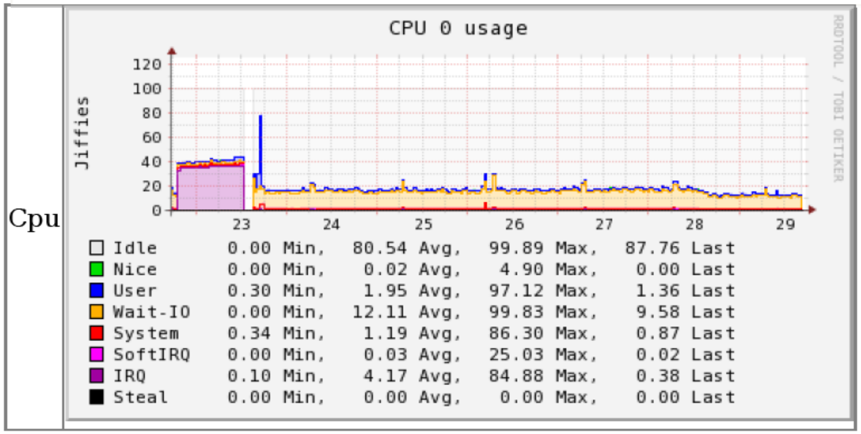
- **Cppcheck** **(sourceforge.net/projects/cppcheck)**
  - Source code analyzed for various software vulnerabilities
  - Report identifies specific lines of code with potential issues
  - Does not require software modifications
  - Benefits
    - FOSS
    - Noted issues are identified and resolved prior to run-time testing
    - Significant reduction in memory leak stability issues during initial software integration efforts

Category	Description
Memory Leaks	The most basic example of a memory leak is when memory is dynamically allocated but not de-allocated when a function terminates.
Resource leaks	Occurs when an acquired resource cannot be released.
Array out of bounds	Occurs when accessing an element of an array using an index that is beyond the size originally allocated for the array.
Buffer overruns	Can occur when the destination of a copy is smaller than the source.
Memset on class	Memset is typically used as a convenient way to set all data members of a class to zero. However, if the class contains any virtual functions, it is likely that the C++ internal virtual pointer table will be overwritten causing the virtual pointer to point to NULL.
Overlapping data buffer	Can occur when specifying the same source and destination for a memory operation
Missing virtual destructors	In object oriented programming, and in this case C++, it is common to use inheritance to define a "derived" class to extend a previously defined "base" class. This derived class may then perform any valid C/C++ instructions, including dynamically allocating memory in the constructor, and then (hopefully) deleting the memory in the destructor. A problem can occur because it is possible to delete/destroy the object using the destructor of the base class. Doing this will not execute the destructor in the derived class and therefore not delete the dynamically allocated memory which causes a leak.
Mismatching alloc/dealloc	Using the "new" operator in C++, it is possible to dynamically allocate memory for a pointer to a single object, and a pointer to an array of objects. In both cases, the "delete" operator is used but the syntax differs between the two types of allocations. In the case of the array, "delete" requires the use of the square bracket operators for proper de-allocation to occur.
Division with signed & unsigned operators	The result of a division will be wrong if one operand is unsigned and the other operand is negative.

Example Output:

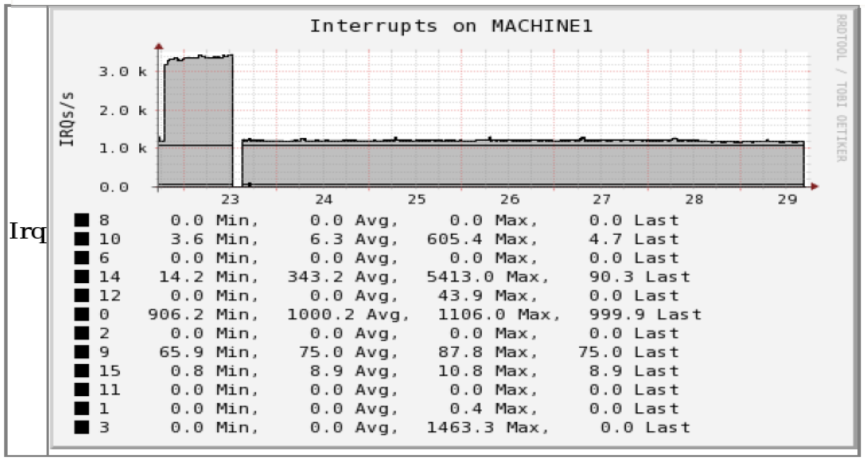
```
FILE         LINE#       ERROR CATEGORY      ERROR  TYPE                          VARIABLE NAME
file1.cpp  :   21             (error)          memory leak             :              test
file2.cpp  :  110             (error)          resource leak           :          class::create
file10.cpp :  204             (error)          overlapping data buffer  :            template
```

- **Collectd** **(collectd.org)**
  - Ability to monitor and analyze CPU, memory and network utilization at the system and element level
    - Synchronized process-to-infrastructure views
    - Per-core and aggregate CPU statistics
  - Non-intrusive capability available on-demand
  - Benefits
    - FOSS
    - Provides critical infrastructure TPM’s
    - Allows detection and investigation of anomalies early in the development cycle, thus ensuring greater stability in later deliveries



```
□ Idle      0.00 Min,  80.54 Avg,  99.89 Max,  87.76 Last
■ Nice      0.00 Min,   0.02 Avg,   4.90 Max,   0.00 Last
■ User      0.30 Min,   1.95 Avg,  97.12 Max,   1.36 Last
■ Wait-IO   0.00 Min,  12.11 Avg,  99.83 Max,   9.58 Last
■ System    0.34 Min,   1.19 Avg,  86.30 Max,   0.87 Last
■ SoftIRQ   0.00 Min,   0.03 Avg,  25.03 Max,   0.02 Last
■ IRQ       0.10 Min,   4.17 Avg,  84.88 Max,   0.38 Last
■ Steal     0.00 Min,   0.00 Avg,   0.00 Max,   0.00 Last
```

Per-Core Utilization

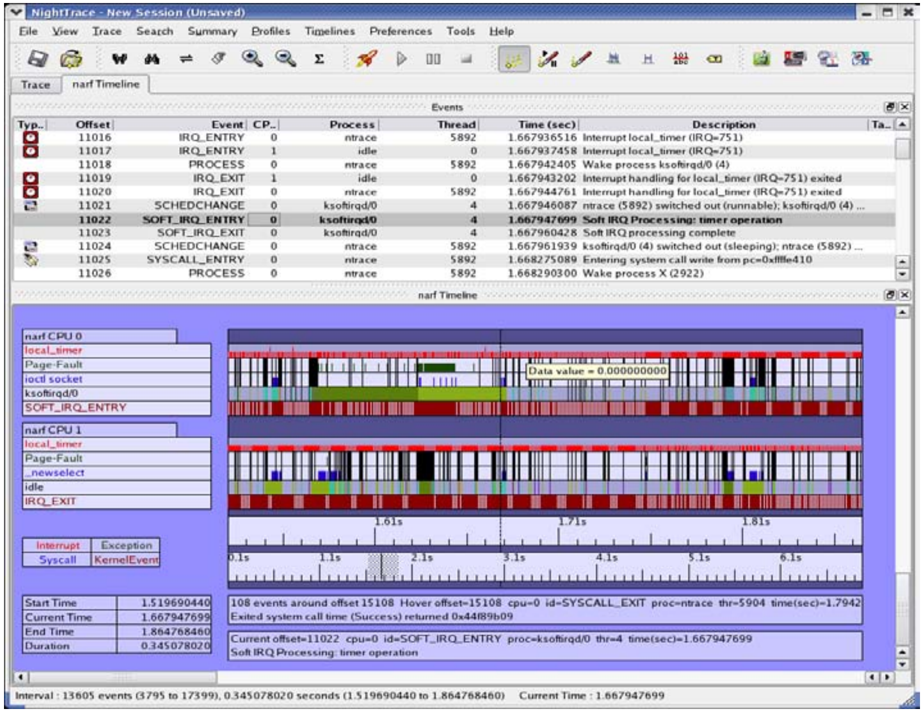


```
■ 8       0.0 Min,     0.0 Avg,     0.0 Max,     0.0 Last
■ 10      3.6 Min,     6.3 Avg,   605.4 Max,     4.7 Last
■ 6       0.0 Min,     0.0 Avg,     0.0 Max,     0.0 Last
■ 14     14.2 Min,   343.2 Avg,  5413.0 Max,    90.3 Last
■ 12      0.0 Min,     0.0 Avg,    43.9 Max,     0.0 Last
■ 0     906.2 Min,  1000.2 Avg,  1106.0 Max,   999.9 Last
■ 2       0.0 Min,     0.0 Avg,     0.0 Max,     0.0 Last
■ 9      65.9 Min,    75.0 Avg,    87.8 Max,    75.0 Last
■ 15      0.8 Min,     8.9 Avg,    10.8 Max,     8.9 Last
■ 11      0.0 Min,     0.0 Avg,     0.0 Max,     0.0 Last
■ 1       0.0 Min,     0.0 Avg,     0.4 Max,     0.0 Last
■ 3       0.0 Min,     0.0 Avg,  1463.3 Max,     0.0 Last
```

Node-Level Interrupts

PE Framework

- **NightTrace** **(Concurrent Computer Corporation)**
  - Deterministic debugging, monitoring, tracing and tuning
  - Ideal for time-critical applications
  - Synchronized graphical or text display of system and application activity
  - Benefits
    - Easy-to-use graphical user interface
    - Provides user-defined event logging
    - Customizable views
    - Correlate kernel events with application activity



Typ..	Offset	Event	CP..	Process	Thread	Time (sec)	Description	Ta..
	11016	IRQ_ENTRY	0	ntrace	5892	1.667936516	Interrupt local_timer (IRQ=751)	
	11017	IRQ_ENTRY	1	idle	0	1.667937458	Interrupt local_timer (IRQ=751)	
	11018	PROCESS	0	ntrace	5892	1.667942405	Wake process ksoftirqd/0 (4)	
	11019	IRQ_EXIT	1	idle	0	1.667943202	Interrupt handling for local_timer (IRQ=751) exited	
	11020	IRQ_EXIT	0	ntrace	5892	1.667944761	Interrupt handling for local_timer (IRQ=751) exited	
	11021	SCHEDCHANGE	0	ksoftirqd/0	4	1.667946087	ntrace (5892) switched out (runnable); ksoftirqd/0 (4) ...	
	11022	**SOFT_IRQ_ENTRY**	**0**	**ksoftirqd/0**	**4**	**1.667947699**	**Soft IRQ Processing: timer operation**	
	11023	SOFT_IRQ_EXIT	0	ksoftirqd/0	4	1.667960428	Soft IRQ processing complete	
	11024	SCHEDCHANGE	0	ntrace	5892	1.667961939	ksoftirqd/0 (4) switched out (sleeping); ntrace (5892) ...	
	11025	SYSCALL_ENTRY	0	ntrace	5892	1.668275089	Entering system call write from pc=0xffffe410	
	11026	PROCESS	0	ntrace	5892	1.668290300	Wake process X (2922)	

Start Time	1.519690440
Current Time	1.667947699
End Time	1.864768460
Duration	0.345078020

108 events around offset 15108 Hover offset=15108 cpu=0 id=SYSCALL_EXIT proc=ntrace thr=5904 time(sec)=1.7942
Exited system call time (Success) returned 0x44f89b09

Current offset=11022 cpu=0 id=SOFT_IRQ_ENTRY proc=ksoftirqd/0 thr=4 time(sec)=1.667947699
Soft IRQ Processing: timer operation

Interval : 13605 events (3795 to 17399), 0.345078020 seconds (1.519690440 to 1.864768460) Current Time : 1.667947699

NightStar Tool Suite

Concurrent Computer Corporation

Provides visibility of application timing issues at the Operating System level

PE Framework

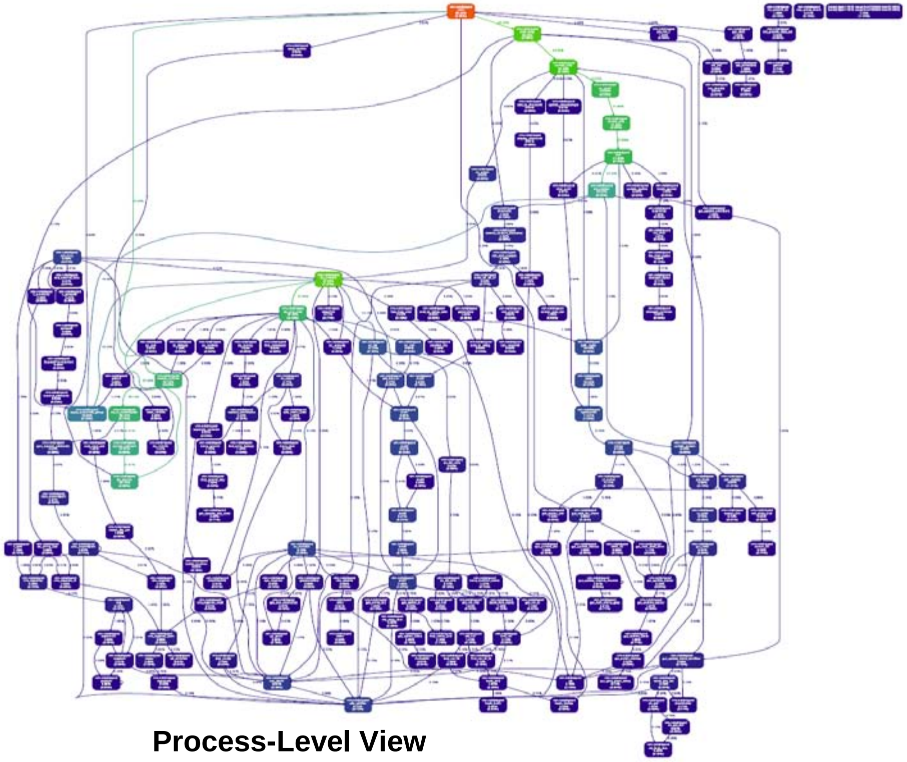
- **Data Recording** **(U.S. Navy - Aegis / LM / MSE)**
  - Utilizing extraction point data to implement a TPM “report card” for system and application timing and capacity requirements,
  - Extraction point data is recorded during all test events
  - Benefits
    - Application instrumentation
    - Ability to toggle EP recording on/off at system consoles
    - Used for timing & capacity measurements, SW studies, and correlation of anomalies

Cornerstones	TPM	Build x	Build y
Availability	CPU Utilization Normal	🟩	🟩
	CPU Utilization Stress	Not Tested	Not Tested
	Memory Utilization	🟩	🟩
	Network Utilization	🟩	🟨
	Initialization	🟩	🟩
Reaction Time	Tracking	Not Tested	🟩
	AAW in Adverse	Not Tested	Not Tested
	AAW in Clear	Not Tested	Not Tested
	Display	🟩	🟨
Firepower	Engagement	Not Tested	🟩
	Firing Rate	Not Tested	Not Tested
	MAX** – AAW	Not Tested	Not Tested
	MAX** – Multimission	Not Tested	Not Tested
Coverage	Track Capacity	Not Tested	🟩

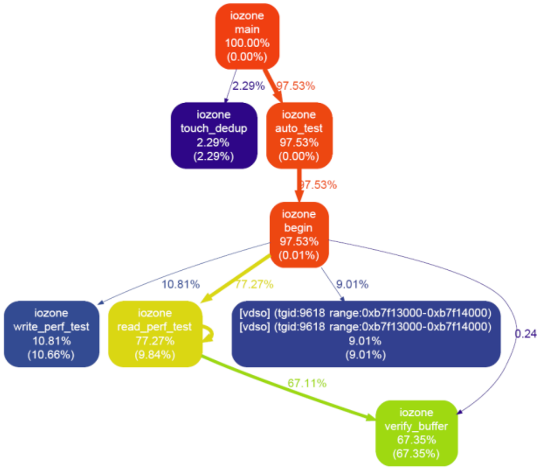
Example TPM Report Card

PE Framework

- **Oprofile** **(sourceforge.net/projects/oprofile)**
  - Shows most frequently executed software threads
  - Very low overhead
  - Benefits
    - FOSS
    - Allows engineers to understand the behavior of the software
    - Reveals excessive processing / loops
    - Identifies areas for further instrumentation



Process-Level View



```
    7  0.1288 :                    xx2=chid;
   25  0.4602 :   if(share_file)
              :                    xx2=(long long)0;
    5  0.0920 :   mpattern=patt;
   10  0.1841 :   pattern_buf=patt;
              :
   10  0.1841 :   where=(unsigned long long *)buffer;
              :
    2  0.0368 :   if(!verify)
              :                    printf("\nOOPS Entered unexpectedly !!! \n");
              :
              :
    4  0.0736 :   if(sverify == 2)
              :   {
              :   }
```

Source-Code Annotation

Summary

U.S. Navy Photo

Adaptable Performance Engineering Initiative

- Highly configurable suite of open source, commercially available, and developed tools that can provide measurements and diagnostics across the enterprise

Development Site

Validation of the SW during development phase

Test Facility

Shipyard Integration

System validation, diagnostics, operability tests

Deployed Systems

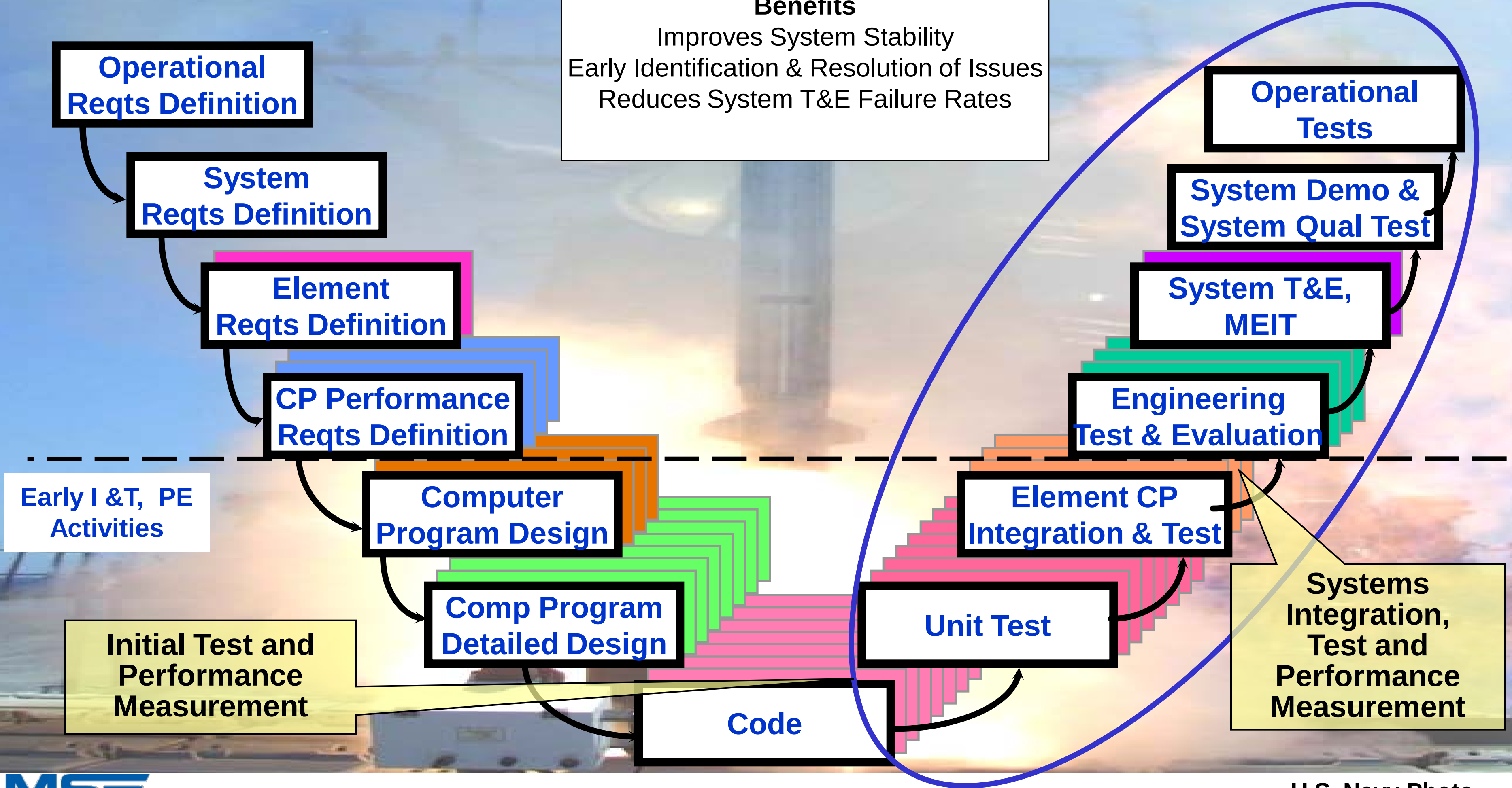
Runtime status monitoring, operability tests, diagnostics

U.S. Navy Photos

Performance Engineering Initiatives

- Increasing our ability to characterize and analyze complex systems in a COTS environment

Benefits
Improves System Stability
Early Identification & Resolution of Issues
Reduces System T&E Failure Rates


Operational Reqts Definition
System Reqts Definition
Element Reqts Definition
CP Performance Reqts Definition
Computer Program Design
Comp Program Detailed Design
Code
Unit Test
Element CP Integration & Test
Engineering Test & Evaluation
System T&E, MEIT
System Demo & System Qual Test
Operational Tests
Early I &T, PE Activities
Initial Test and Performance Measurement
Systems Integration, Test and Performance Measurement

U.S. Navy Photo

Contact Information

Edward Beck
Sr. Manager
Applied Integration Solutions

MSE, LLC
304 West Route 38
Moorestown, NJ 08057

(856) 252-2055
ebeck@missionse.com

U.S. Navy Photo

Tool Reference (Origin)

Tool	Description	Organization
cppcheck	Performs static analysis of C/C++ source code for software vulnerabilities	Sourceforge/projects/cppcheck
valgrind	Performs run-time analysis of executables for instances of memory leaks	Valgrind.org
collectd	Provides dynamic display of system and application performance – cpu, memory and network utilization. Historical data is available for trend analysis.	Collectd.org
ntop	Provides run-time data on network utilization	Ntop.org
NightTune	Provides dynamic display of system and application performance – cpu, memory and network utilization. Historical data can be correlated with NightTrace data-set for TPM analysis	Concurrent Computer Corporation
NightTrace	Provides synchronized graphical or text display of all system activity	Concurrent Computer Corporation
Data Recording	Provides analysis of tactical run-time extraction point data	U.S. Navy / LM / MSE
oprofile	System-wide profiler for Linux systems, capable of profiling all running code at low overhead	Sourceforge/projects/oprofile
wireshark	Provides run-time analysis of detailed network packet information	Wireshark.org