Carnegie Mellon
Software Engineering Institute
Pittsburgh, PA 15213-3890

Diagnostic Software

What your Developer Doesn't Know

Ted Marz
tfm@sei.cmu.edu

A presentation of paper CMU/SEI-2005-TN-035
Integrated Diagnostics: Operational Missions, Diagnostic Types, Characteristics, and Capability Gaps
http://www.sei.cmu.edu/publications/documents/05.reports/05tn035.html

Sponsored by the U.S. Department of Defense

Motivation

- Involved in several software intensive systems development activities
- Observed a lack of operational knowledge on diagnostics in the system development teams
- Lack of knowledge in non-traditional developments
- Near total lack of integration between O-Level and I-/D-Level diagnostic and repair activities
- Seen how diagnostics can impact Life Cycle Cost
  - Increased Spares
  - CND / RTOK rates in the repair process
  - Manning / Staffing issues of operational systems

Diagnostic Software

The DoD is dependent on increasingly complex, software intensive, hardware/software hybrid systems to achieve their mission.

Assurance of mission capability is a primary operational need.

- Fault Detection (FD) supports that need
- Fault Isolation (FI) assists in assessing the impact of a failure

Diagnostic capabilities are a co-development problem.

Lack of effective FD/FI and Restoration practices impact system lifecycle cost in multi-dimensional ways.

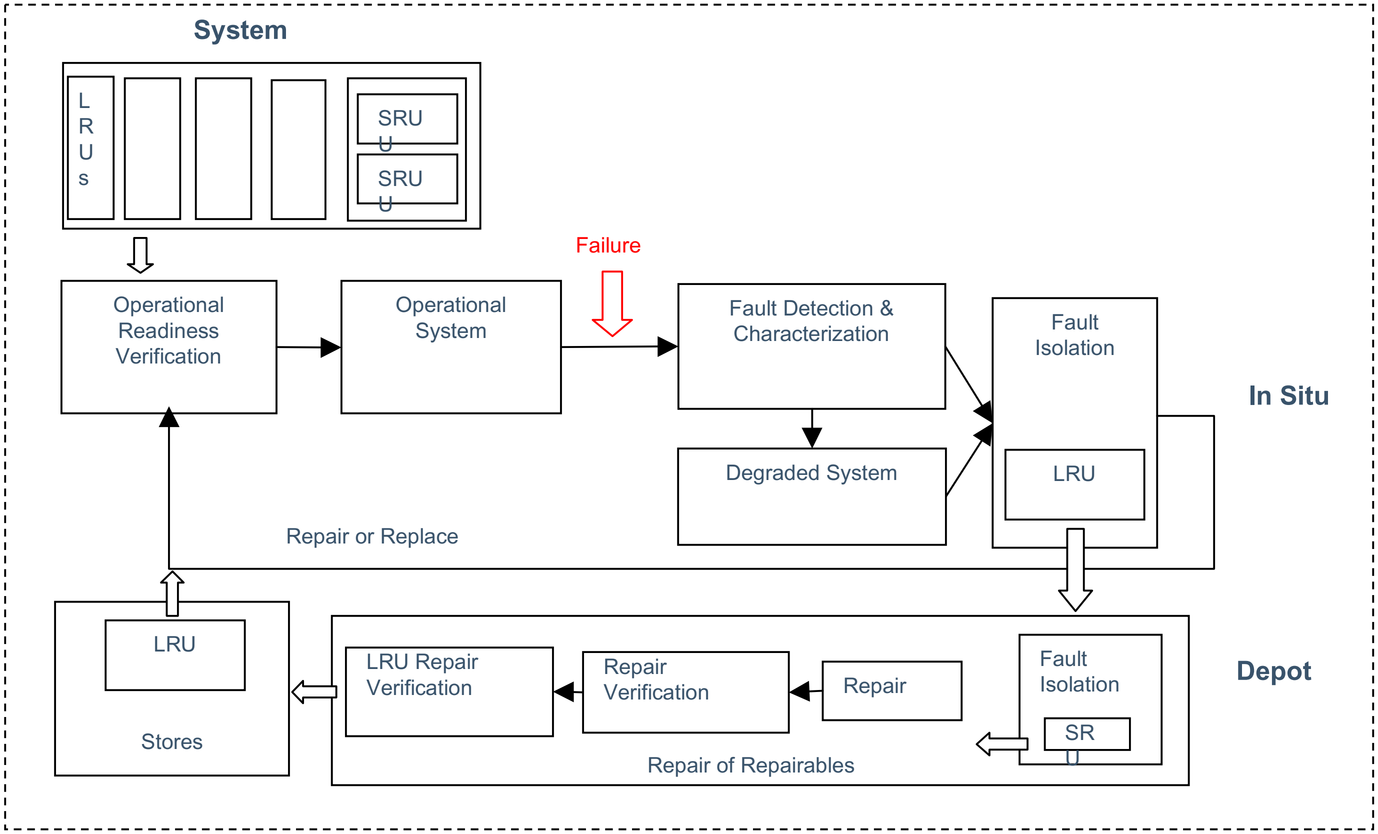
FD/FI capabilities are not generally considered core requirements by the developers.

Diagnostic Operational Missions

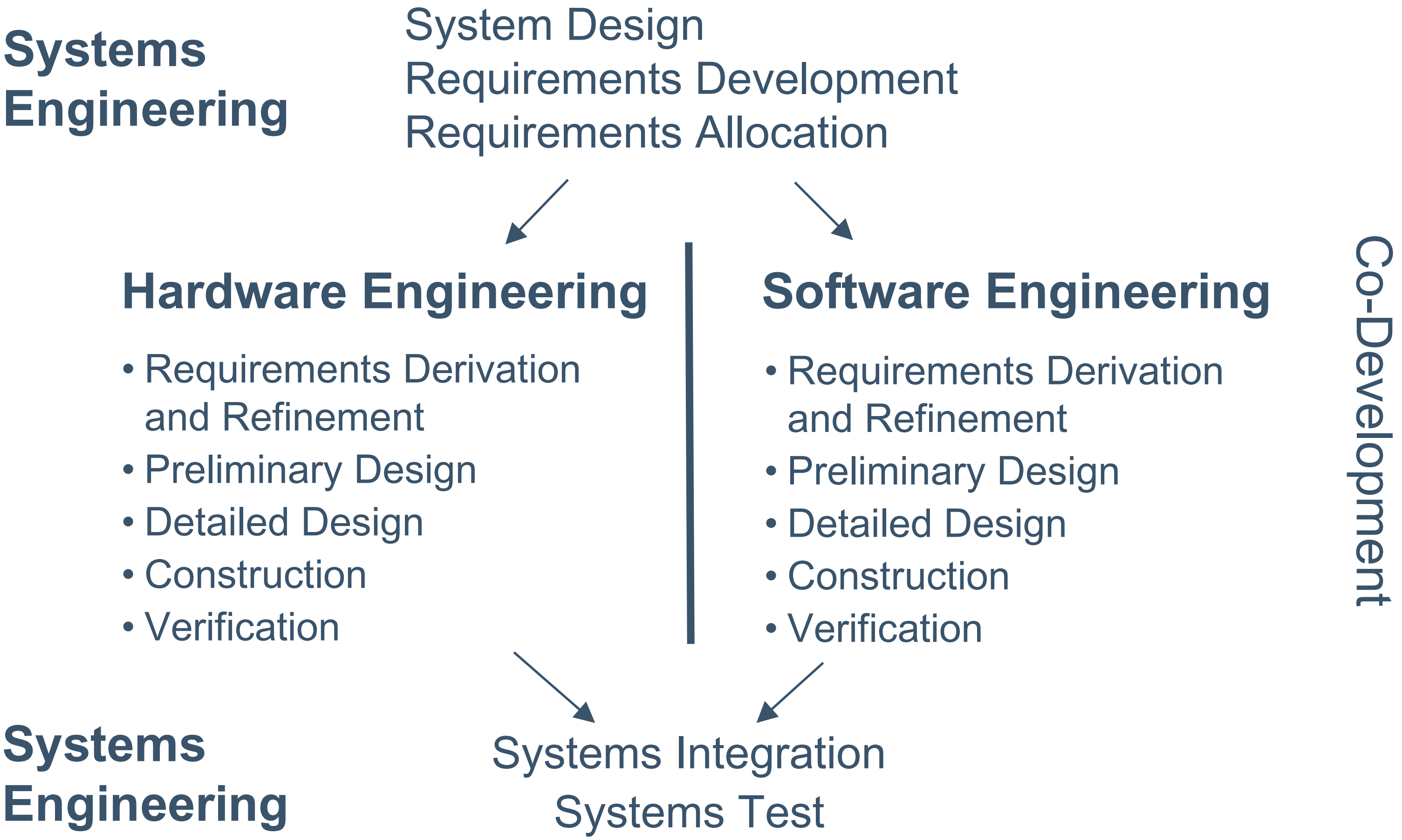
- Verification of Operational Readiness
  Am I Mission Capable?
- Fault Detection (FD) and Characterization
  Have I failed mid-mission?
  What are the effects of failure? Can I continue?
- Fault Isolation (FI)
  What has failed? What do I need to replace?
- Diagnosis and Repair of Repairables
  FI at the lower component level; Repair verification
- Other Maintenance Actions
  Installation, Configuration, Alignment, Calibration, etc.

Carnegie Mellon
Software Engineering Institute

Logistics Support Cycle



System Development Process



Systems Engineering

System Design
Requirements Development
Requirements Allocation

Hardware Engineering

- Requirements Derivation and Refinement
- Preliminary Design
- Detailed Design
- Construction
- Verification

Software Engineering

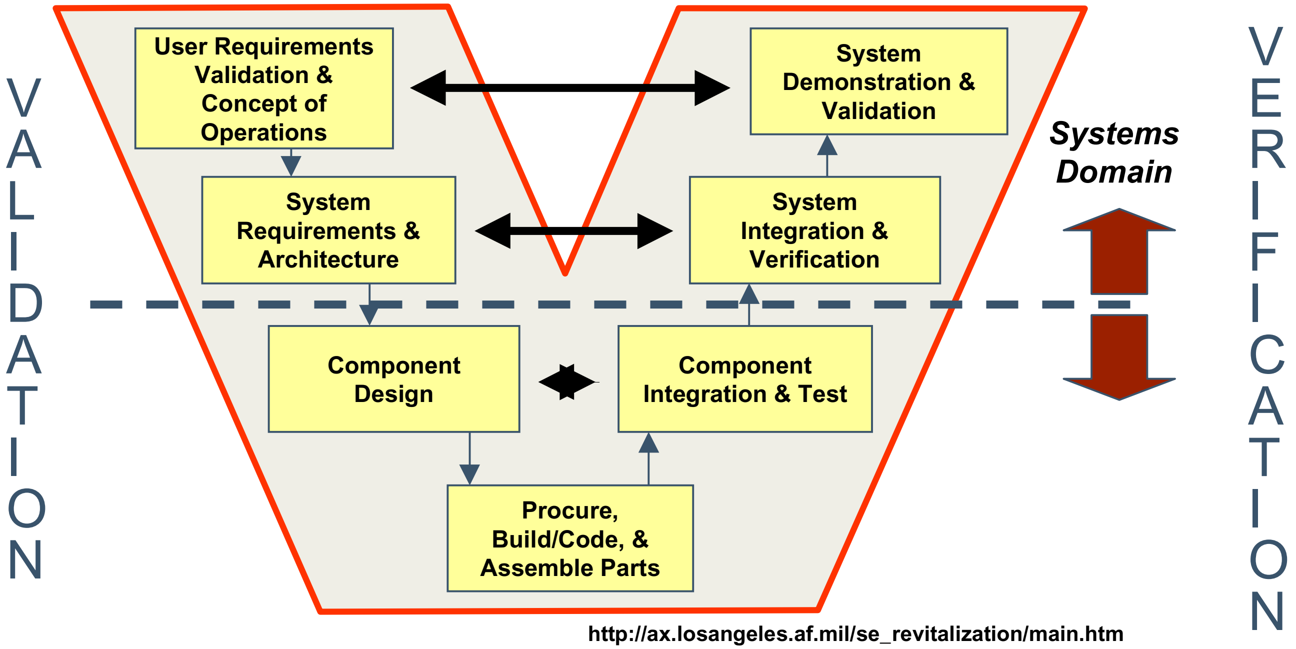
- Requirements Derivation and Refinement
- Preliminary Design
- Detailed Design
- Construction
- Verification

Co-Development

Systems Engineering

Systems Integration
Systems Test

System Validation Activities


VALIDATION
User Requirements Validation & Concept of Operations
System Requirements & Architecture
Component Design
Procure, Build/Code, & Assemble Parts
Component Integration & Test
System Integration & Verification
System Demonstration & Validation
Systems Domain
VERIFICATION

http://ax.losangeles.af.mil/se_revitalization/main.htm

- Engineering Reviews at all levels are Validation events
- Acquisition Program Office MUST participate in validation events.
  - Balanced with other responsibilities
  - Resourced with appropriate capability

System Safety influence diagnostic maturity

Safety is a prime driver, as it is a major concern of the verification and validation efforts.

Domains with strong safety concerns exhibit more mature diagnostic environments

- Regulatory & Liability responsibilities drive activities
- System Safety Engineering Program
  - Failure Modes, Effects & Criticality
  - Undiagnosed failures lead to unsafe conditions
  - Recognized software safety standards applied

Example Domains

- Avionics & Flight controls
- Nuclear & other Power Generation
- Chemical Process Control
- Medical Instrumentation & Devices
- Telecom

Even Mature Environments Fail

Example – recent F-22 flight controls related crash.

Non-Traditional Environments Fail Spectacularly

Example – mission critical IT system

No verification of operational readiness
No online fault detection / isolation
Internet hosting service not doing system performance monitoring

Hardware BIT is not sufficient

Diagnostics is an Operational Mission need

- Verify capability wherever it is implemented
  - Distributed, “Net Centric” & SOA systems
  - Programmable Hardware environments (FPGA, etc.)
  - Software implemented capabilities
- Software component health has not been a significant concern to date
  - Ad Hoc methods
  - Spotty coverage
  - Inconsistent handling & reporting
- Software health reporting should be part of the overall systems health management environment

What Developers Should Do

- Consider the Integrated Diagnostics and other System Sustainment and Support capabilities part of the core mission
- Explicitly treat Integrated Diagnostics as a co-development problem, with appropriate, multi-disciplinary Integrated Product Team support
- Fold software health management into the overall system health management environment
- Better consider integration of the in-situ and Depot diagnostics environments

What Program Offices Should Do

- Better integrate logistics support (diagnostics, test, maintenance, repair) in the development activities currently supported by the Hardware and Software validation teams
- Resource the validation teams to better support the acquisition effort
  - Be prepared to augment the developer with operations expertise from similar, legacy systems
- Create realistic diagnostic coverage requirements
- Better define the needs of the on-line and off-line diagnostics environments
- Create requirements for the integration of the in-situ and Depot maintenance environments

Carnegie Mellon
Software Engineering Institute

Contact Information

Ted Marz tfm@sei.cmu.edu