Delivering Capabilities through Requirements Management

1 Abstract

Requirements are in terms of capabilities. Large productivity gains have been demonstrated using a targeted capability model on some of the most complex systems. Core engineering process issues are addressed from the perspective of the low level developer and architect. Emphasis is placed on risk, accountability, and performance.

1.1 Identification

March 15, 2012, Chicago INCOSE.
Presenter: Ty Zoerner, Infinite Delta Corp, `www.InfiniteDelta.com`
`ty@InfiniteDelta.com`

$Id: electronic_capabilities_management.tex 40 2012-03-09 02:16:42Z ty $
$HeadURL: svn+ssh://InfiniteDelta.com/svn/papers/electronic_capabilities_management/electronic_capabilities_management.tex $

1.2 Apollo 13 Movie Quote

Apollo 13 Movie: Eugene Francis "Gene" Kranz, NASA Flight Director: "I don't care about what anything was **DESIGNED** to do, I care about what it **CAN** do."

Contents

2 Introduction

- Turning of the tide, renewed emphases on:
  - Cost, Schedule, and Risk in Avionics.
  - Proactive Capabilities Process
  - Genuine Quality (Robert Pirsig)
  - Human Factors
- Requirements are in terms of Capabilities (tightly coupled).
- History and lessons learned from a low level perspective.
- Engineering Issues
- Development and process models.
- Risk, Accountability, and Performance.

3 History

- Navy P-3 Aircraft Crash
- Technical Domains and Management
- Avionics Distributed Database
- Conveyor Emulation System
- Avionics Test System
- Business models and planned obsolescence.

3.1 Lockheed P-3 Orion

Anti-submarine and surveillance aircraft. (1961-present)

3.2 P-3 Crash and DO-178B safety

- P-3 crash into a mountain in 1977, Canary Islands. Crew of 13 lost.
- DO-178B Safety Levels

  A. aircraft cannot continue to fly or land.
  B. major safety concern, people are at risk.
  C. mission is at risk enough to affect safety.
  D. mission is at risk.
  E. cannot affect safety, and should not affect mission.

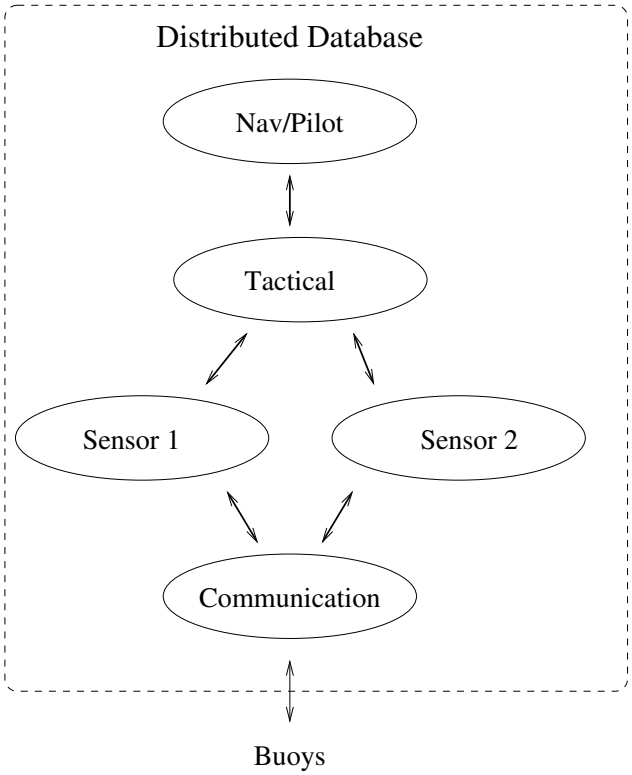
- Technical domain team leaders include the Navy officers.
- Human Factors are critical for engineering and flight crew.

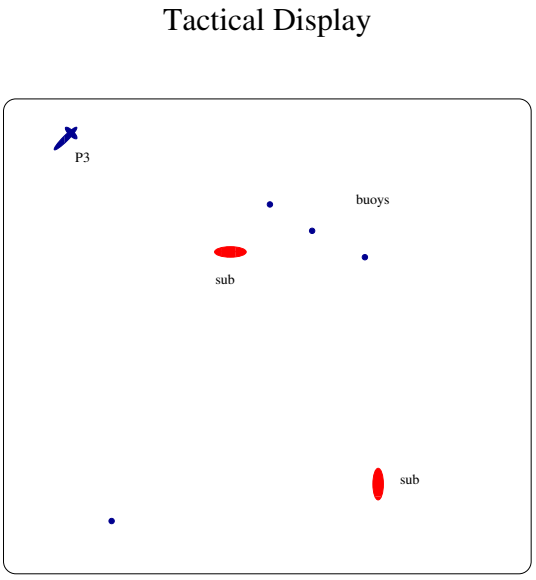
3.3 Technical Domains and Management

- Each technical domain is held fully accountable (cost, schedule, and risk).
- Bids required sign-off of each technical domain.
- Requirements were only written after knowing the capability and its restrictions.
- Core system requirements and design where complete before the bids committed.
- Business and engineering objectives are in full alignment.
- Technical ownership is the foundation of this process (Robert Pirsig's Quality).
- Effective control is in the technical domain.
- Failure analysis includes the engineering process.

3.4 P-3 Avionics Distributed Database

Avionics Distributed Database for five major subsystems:

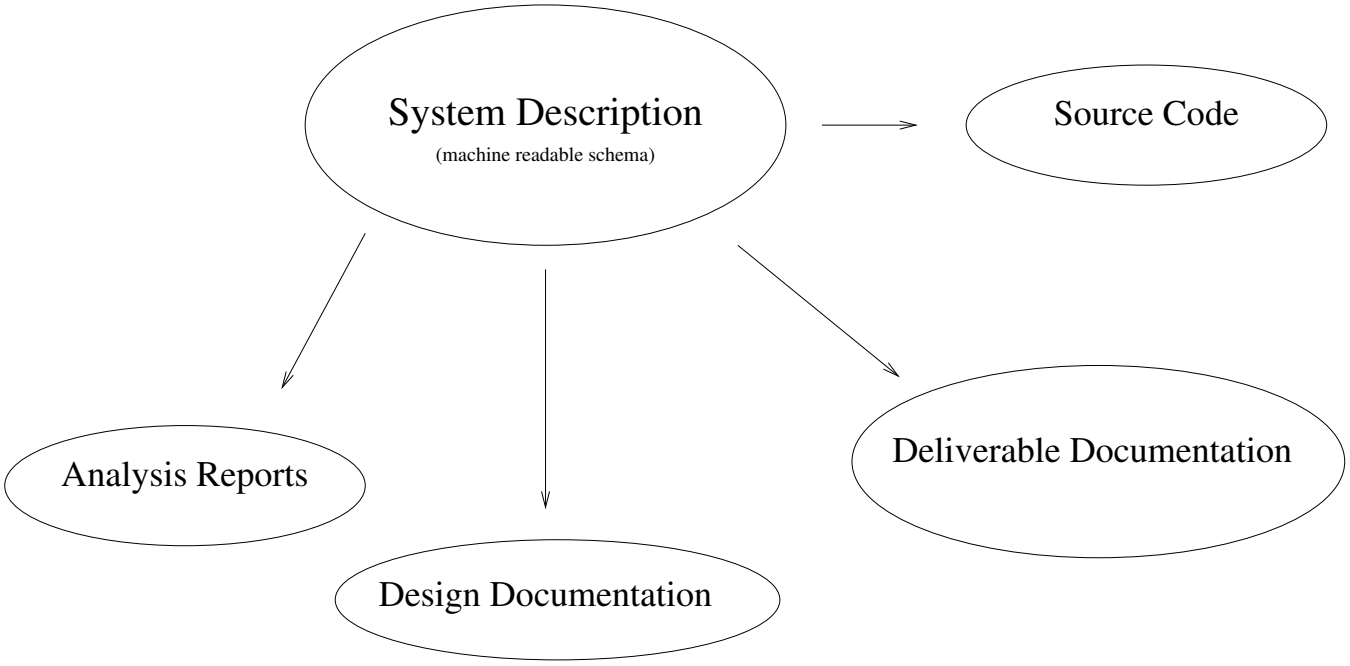




$Id: ddblayout.fig 32 2012−03−04 21:58:12Z ty $

$HeadURL: svn+ssh://InfiniteDelta.com/svn/papers/electronic_capabilities_management/ddblayout.fig $

3.4.1 Distributed Database Process



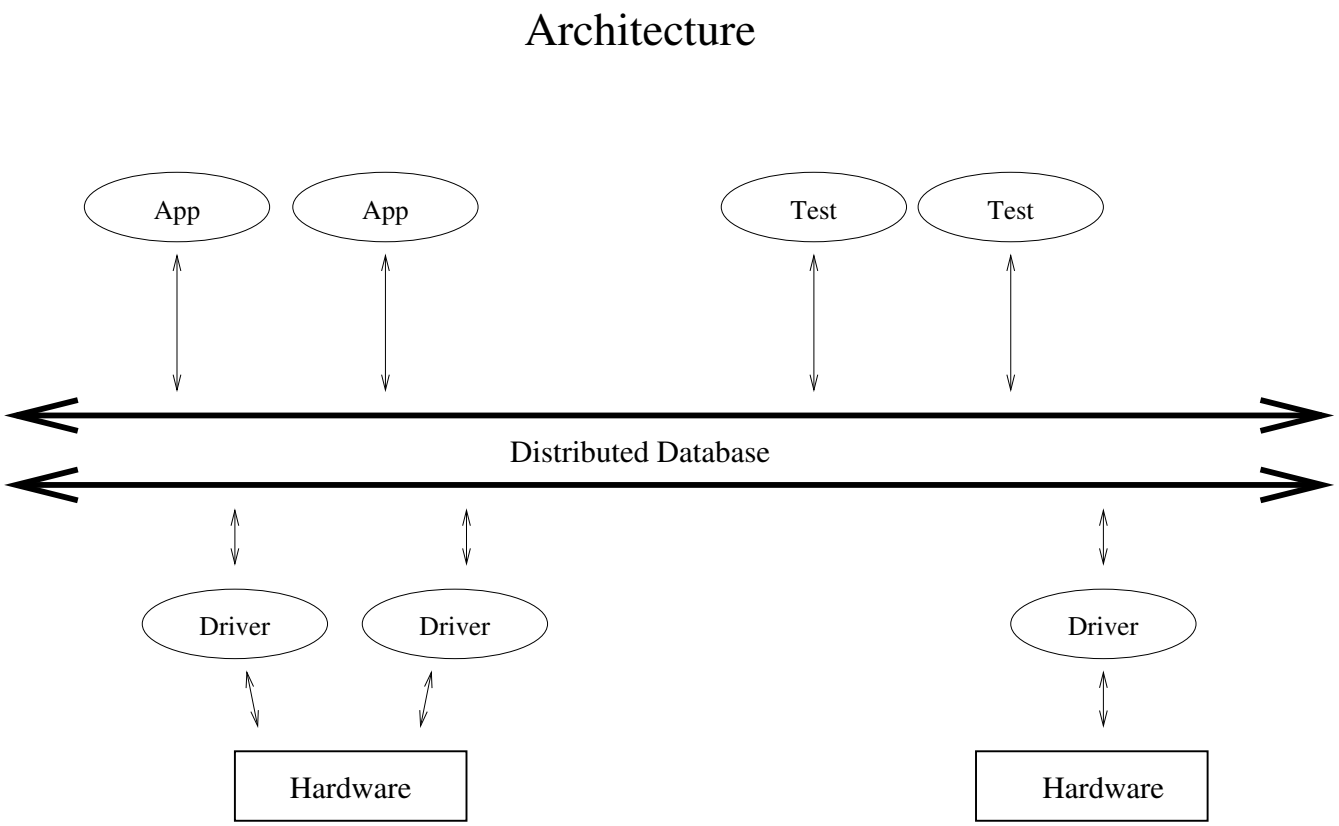
$Id: ddbschema.fig 29 2012−03−01 09:35:58Z ty $

$HeadURL: svn+ssh://InfiniteDelta.com/svn/papers/electronic_capabilities_management/ddbschema.fig $

3.4.2 Capabilities for ALL!

- The specification is fully machine readable.
- Verified for accuracy and restrictions.
- Produced critical code and documentation.
- Minimized man power and integration times.
- Demonstrated maximum operational capabilities.
- Highest degrees of group and version independence.
- System level changes where easy, low impact, and nearly risk free.
- A major feature was committed to the customer with confidence.

3.4.3 Architecture



$Id: ddbarch.fig 33 2012−03−05 12:00:25Z ty $

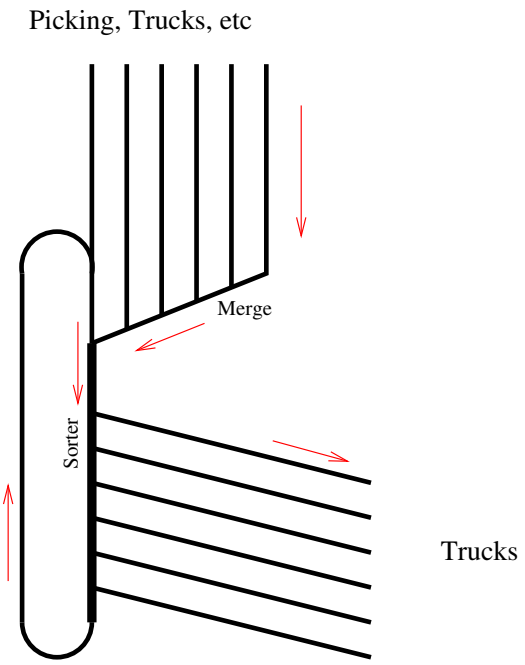
$HeadURL: svn+ssh://InfiniteDelta.com/svn/papers/electronic_capabilities_management/ddbarch.fig $

3.4.4 Results

- Generated analysis, critical source code, and documentation.
- Averaged three changes a day for the first six months.
- Software applications are now tightly coupled with capabilities.
- Systems now has a cafeteria menu, full of capabilities to be selected.
- Systems is in the driver's seat, allowing the software groups to run.
- Software had necessary resources to meet their commitments.
- Within the first two weeks of integration, 85 percent of the system was operational.

3.5 Package and Food Material Handling System

Typical conveyor layout:



$Id: conveyor.fig 28 2012−02−29 17:08:07Z ty $

$HeadURL: svn+ssh://InfiniteDelta.com/svn/papers/electronic_capabilities_management/conveyor.fig $

General shippers (RPS, FedEx, etc.), food, and general warehouse material handling.

3.5.1 Conveyor Emulation System

- Electronically readable conveyor system specification.
- The specification was another "Application Specific Language" (ASL).
- Verified for accuracy and restrictions.
- Produced operational test system.
- Minimized test architect effort.
- Certified communication protocols.
- Demonstrated advantages of Weighted Requirements.
- Found major holes in the Modeling System used for validation.

3.6 Avionics Hardware Test System

- A new PowerPC product needed to be loadable and pre-screened before given to software.
- PowerPC product lined needed a General Purpose Monitor (GPMon).
- Monitors are self-contained, stand-alone boot programs, that allow basic diagnostics and loading of other applications.
- GPMon is an Ethernet based Monitor, able to run RAM based applications.
- GPMon development was on Linux using GCC configured for embedded cross platform development.
- Hardware needed confirmation of problems found by software.
- Hardware performance testing required a full tickless real-time kernel.
- GPMon prescreening and loading is now used on four PowerPC based products.

3.6.1 C130 AMP Chassis

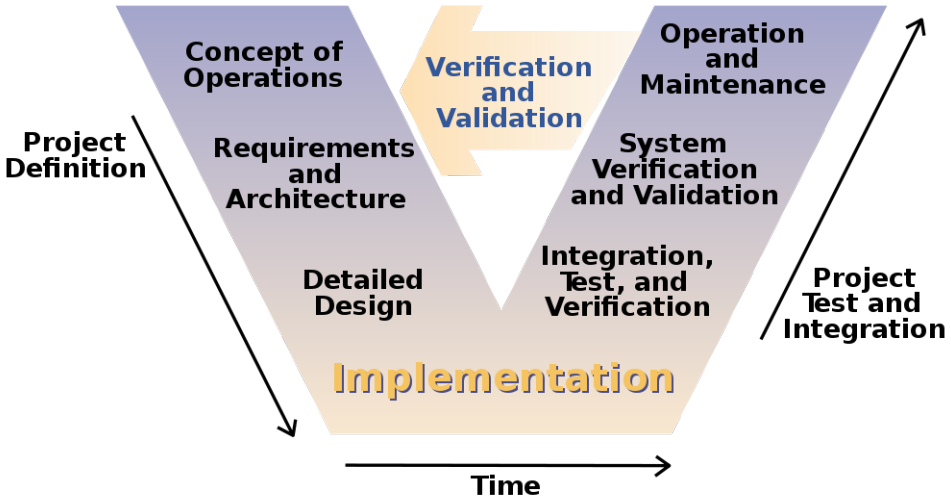
Initial C130 AMP chassis delivery to Boeing's lab:

- System included 20 PowerPC CPUs, 60 ARINC 429, 10 RS422, 20 Ethernet, 8 1553 buses, 20 Discretes, 10 PCI buses and one VME bus.
- Existing test infrastructure organized around the capability model.
- Pre-validated test executables already supported specific tests.
- We need only to verify the test configuration by breaking each wrap.
- Start to finish took four days to create the chassis test.
- The test report was auto-generated using the same LaTeX document preparation system as this presentation.
- The GPMon development infrastructure was selected for Boeing's 787 core computer robustness testing.

3.7 Alignment of Business and Engineering Objectives

- Cost and schedule objectives.
  - Technical ownership and accountability are paramount. (Robert Pirsig's Quality)
  - System risk was identified and well quantified.
  - Architecture battles where upfront and technical.
  - Lost contracts to under bidding competitors.
- Under bidding, planned overruns, vendor model.
  - Best guess bidding creates a reactive development model.
  - Profit center moved from cost/schedule to time/material.
  - Reactive man-power intensive model is replacing the proactive capability development.
  - Overhead is reaching record levels.
  - Technical ownership and accountability are devalued.

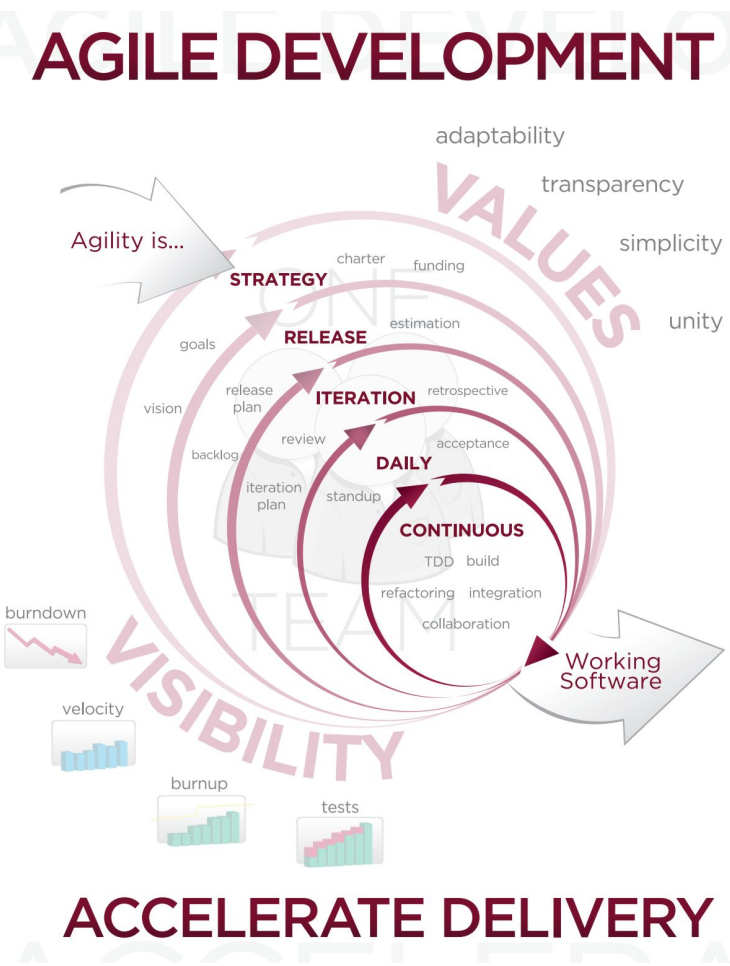
3.8 Development and Process



http://en.wikipedia.org/wiki/V_model

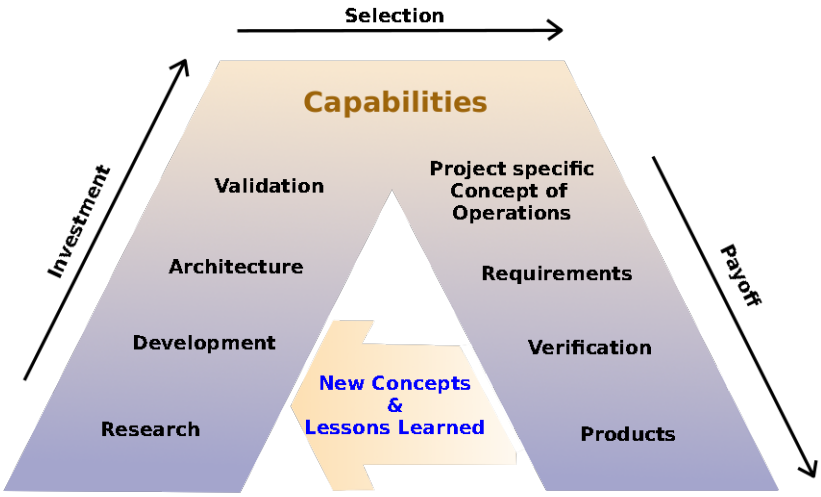
Popular systems V-Model development.

3.8.1 Agile Development



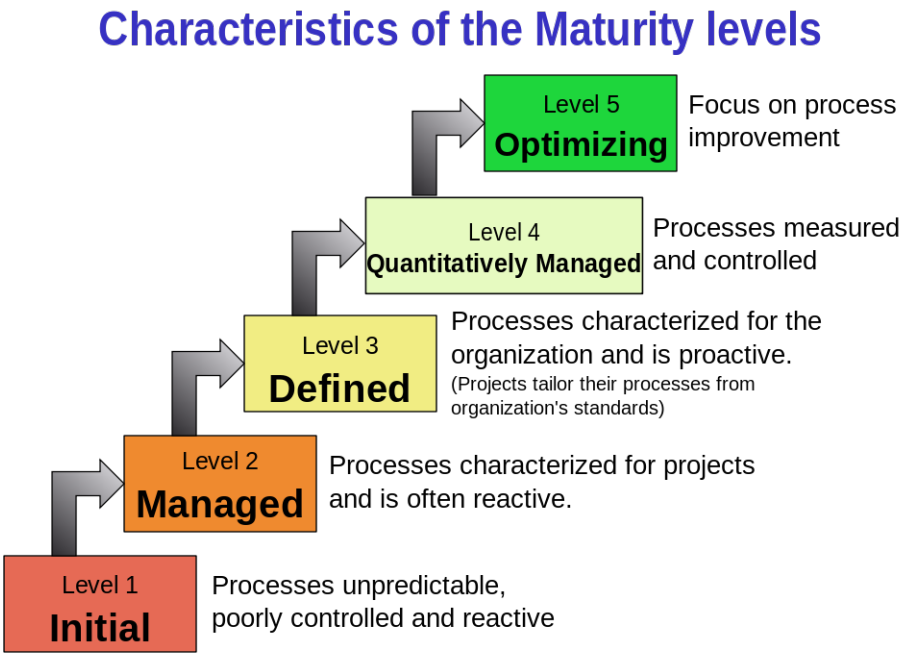
http://en.wikipedia.org/wiki/Agile_software_development

3.8.2 Traditional Capabilities Development (A-Model)



Capabilities Development Model
Looks familiar to the V-Model 3.8.

3.8.3 Capability Maturity Model Integration



`http://en.wikipedia.org/wiki/CMMI`

Can you have a proactive process with a reactive development environment?

3.8.4 Integrated Modular Avionics (IMA) Architecture

- Multiple applications on a computer, often from different vendors.
- Multiple DO178B safety levels.
- Incremental acceptance of applications on a reference platform.

3.8.5 Development Summary

- V-Model: Top down requirements with feedback.
  - Currently acceptable development approach.
  - Logarithmic development schedules and costs.
  - Mostly from feedback and corrections.
  - Repeating effort and lost ownership.
- Agile: iterative prototype to production.
  - Becoming very popular.
  - Tightly coupled with customer.
  - Re-engaging engineer's ownership.
  - Higher risk of hitting a dead end.
- A-Model: Capabilities Development
  - Old school with the highest ownership by engineers.
  - Places capabilities above requirements and products.
  - Most bad requirements are preventable.
  - Higher initial investment, but with lower total costs.
  - Linear development with tighter accountability and risk control.
  - The return of innovation.

3.9 Management and System Engineering Issues

As systems grow in size or complexity we need objective:

- Big picture visibility and timely impact analysis.
- Performance analysis of the development, including equipment, personnel, and vendors.
- Available options and potential solutions.
- Analysis when possible comes from an automated source.
- Analysis of data and development security.

3.10 Risk

- Human error is significantly the highest risk factor in all areas, and especially development.
- Technical domains including reliability, quality, safety, human factors, legal, etc., are key.
- Loading, attention, ownership are critical in all stages of use and development.
- Some metrics actually increase risk, mostly by adding load, diverting attention, and devaluing ownership.
- Avionics considers obsolescence a product risk.
  - Hardware parts not being available.
  - Depending on vendors with "Planned Obsolescence" business model.
  - Test and development environments no longer supported.

4 Open Source Software (OSS)

Question: Would you consider open source and open standards?

- Common and stable infrastructure and distribution.
- Nearly indefinite long life support, huge legacy software base.
- GNU's General Public License
- GNU's Lesser General Public License
- GNU's Compiler Collection (GCC)
- Embedded Development
- Debian GNU Linux
- Android (Google/GNU/Linux) advancing open standards.

4.1 Electronic Capabilities Management (ECM)

- Is there any interest in:
  - Re-exploring the A-Model/capabilities centric solution?
  - An Open Source/Open Standard Electronic Capabilities Management (ECM)?
- Current developing ECM with the following features:
  - Machine readable method of maintaining capabilities.
  - Requirements are always stated in terms of Capabilities.
  - Restrictions and traceability are built into the system.
  - System configuration details and documentation are generated.
  - Full project visibility including cost and schedule.
  - Can be embedded within:
    * Software source files, even within Doxygen comments.
    * Hardware VHDL source files.
    * Hardware schematic files.
    * Hierarchical system description files.

4.1.1 ECM goal: Preventing Bad Requirements

- Key resource and capability dependency checking.
- Common infrastructure, not necessarily a common language.
- Distributed repository and key data.
- Support for custom utilities for checking and modeling.
- Strong artifact tracking and support.

4.2 Yet Another Distributed Database (YADD)

ECM is built on top of YADD:

- Engineering distributed database
- Highest functionality in degraded mode
- Designed to support embedded systems
- Contains many key relational features
- High performance with minimal use of character strings
- Multiple Levels of Security (MLS) / Need to Know
- Peer-to-peer network architecture

4.3 Integrated Modular Avionics (IMA) RTOS

Question: Interest in an open source Integrated Modular Avionics (IMA) Real-Time Operating System (RTOS)?

- A tickless prioritized preemptive kernel with full process and driver memory management protections.
- Certified that all applications and drivers are space constrained.
- Certified that all applications are time restrained.
- Certified that all critical applications have their specific resources on each system configuration.
- Certified to support multiple levels of security (MLS).

`http://www.infinitedelta.com/wp/avionics_rtos.pdf`

4.4 Summary

An Electronic Capabilities Management can minimizes cost, schedule, and risk:

- Where possible, move effort to the computer away from humans.
- Delivers maximum capabilities.
- Requires minimal support.
- Eliminates staff time (central architects, etc.).
- Supports both inter- and intra-business cooperation.
- Advertises capabilities.

5 Conclusion

Sometimes you cannot beat a movie quote:

5.1 Captain Jack Sparrow

The only rules that really matter are these:
what a man can do and what a man can't do.
For instance, you can accept that your father was a pirate and a good man or you can't. But pirate is in your blood, boy, so you'll have to square with that someday.
And me, for example, I can let you drown, but I can't bring this ship into Tortuga all by me onesies, savvy?
So, can you sail under the command of a pirate, or can you not?

5.2 References

- Metaphysics of Quality: http://en.wikipedia.org/wiki/Robert_Pirsig.
- Flying Cheap: http://www.pbs.org/wgbh/pages/frontline/flyingcheap/etc/script.html
- DO178B: http://en.wikipedia.org/wiki/DO178B
- Dr. Tom Herald ‘‘Affordable Architectures’’, Oct, 2011 http://incose.org/chicagoland/library.aspx
- Dr. Jennifer Narkevicius, ‘‘Human Factors Tutorial’’, Dec 3, 2011.
- http://en.wikipedia.org/wiki/Single_event_upset
- en.wikipedia.org/wiki/Vasa_(ship)
- INCOSE 2007, Model Based Systems Engineering, http://www.incose.org/enchantment/docs/07docs/07jul_4mbseroadmap.pdf Sanford Friedenthal, Regina Griego, Mark Sampson
- Integrated modular avionics (IMA) http://en.wikipedia.org/wiki/Integrated_modular_avionics http: //ftp.rta.nato.int/public//PubFullText/RTO/EN/RTO-EN-SCI-176///EN-SCI-176-04.pdf
- Migrating Infrastructure to GNU/Linux http://www.infinitedelta.com/wp/going_linux.pdf
- Unrestricted Open Source Software http://www.debian.org/intro/free