

Architecture Round Table

INCOSE Chicagoland 3-18-2010

Russell Kubycheck

SE Contractor – Hamilton Sundstrand

INCOSE Chicagoland Dir-at-Large



Agenda

- Definitions
- Bit of Theory
- Enterprise Systems
- Arch Frameworks
 - Zachman
 - DoDAF
- ARP4754 Architectures
- Examples



Definitions

- **System:** Reichtin defines a system as “a set of different elements so connected or related so as to perform a unique function not performable by the elements alone”
- **Architecture:** the fundamental organization of a system embodied in its components, their relationships to each other and to the environment and the principles guiding its design and evolution.
- **Framework:** A structured method/view for organizing information that defines a system or system of systems.



Bits of Architectural Method Theory

Decisions: System Architects transform a set of needs and goals into an architecture for a system. They do this by *making decisions*. But, the relationship between these decisions and the representation of the architecture is generally *not explicit*. An architecture does not specify a detailed design, it specifies a category of acceptable designs.

Problem space size: Humans are better at “creative thinking and describing relationships between small numbers of parts of the system”.

From an engineering perspective, **modularization** has three main purposes:

1. To make complexity manageable;
2. To enable parallel work; and
3. To accommodate future uncertainty.

*“The greatest leverage in architecting is at the **interfaces**.”*, Eberhardt Rechtin,
Director & Architect of NASA’s Deep Space Network



Where/When to start?

- Someone should start at the top. It most likely will not be you, but an Enterprise policy maker who sets the goals.
 - The policy maker defines needs of an entity be it the US President, DOD or Corporate COE.
- The architectural decomposition of a product can only be started after the requirements of the product/system are identified.
 - This goes for the design of the product/system as well.



Architectural Frameworks

- To help organize and define relationships of information in arch framework in a consistent manner.
- IAF Information Arch Framework – use to capture the meta-data to support decisions



Enterprise Architecture (or SOS)

- Enterprise
 - any organization or group of organizations that has a common set of goals or principles or a single bottom line
- Architecture
 - the structure of components, their relationships to each other and to the environment, and the principles guiding the design and evolution of the entity they describe,** whether that entity is an organization, a system, or a functional or mission area

**** DoD Architecture Framework Version 1.5, Volume I: Definitions and Guidelines, 23 April 2007**



Zachman Framework

- The **Zachman Framework** is an [Enterprise Architecture framework](#), which provides a formal and highly structured way of [viewing](#) and defining an enterprise. It consists of a two dimensional classification matrix based on the intersection of six communication questions (What, Where, When, Why, Who and How) with six rows according to [reification](#) transformations .[\[1\]](#)
- The Zachman framework is not a [methodology](#) in that it lacks specific methods and processes for collecting, managing, or using the information that it describes.[\[2\]](#) The Framework is named after its creator [John Zachman](#), who first developed the concept in the 1980s at [IBM](#). It has been updated several times since.[\[3\]](#)
- The Zachman "Framework" is a [taxonomy](#) for organizing architectural artifacts (in other words, design documents, specifications, and models) that takes into account both who the artifact targets (for example, business owner and builder) and what particular issue (for example, data and functionality) is being addressed.[\[4\]](#)



Zachman Framework Model

	Why	How	What	Who	Where	When
Contextual	Goal List	Process List	Material List	Organizational Unit & Role List	Geographical Locations List	Event List
Conceptual	Goal Relationship	Process Model	Entity Relationship Model	Organizational Unit & Role Rel. Model	Locations Model	Event Model
Logical	Rules Diagram	Process Diagram	Data Model Diagram	Role relationship Diagram	Locations Diagram	Event Diagram
Physical	Rules Specification	Process Function Specification	Data Entity Specification	Role Specification	Location Specification	Event Specification
Detailed	Rules Details	Process Details	Data Details	Role Details	Location details	Event Details



DoD. Arch. Framework (DoDAF)

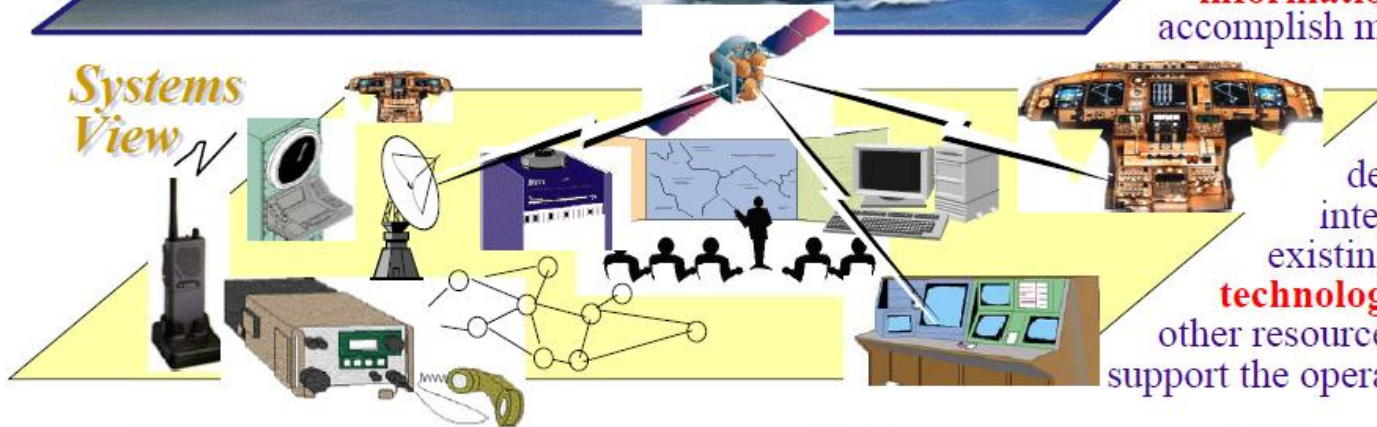
- **Defines a common approach for DoD architecture description, development, presentation, and integration for both war fighting operations and business operations and processes**
 - DoD moves toward net-centric operations and warfare
- **Frame work is intended to ensure that architecture descriptions can be compared and related across organizational boundaries, including Joint and multi-national boundaries**
- **Three related views of architecture:**
 - **Operational view (OV)**
 - **Systems View (SV)**
 - **Technical Standards (TV)**



One Architecture – Three Views



The *Operational View* describes and interrelates the **operational elements**, **tasks** and **activities**, and **information flows** required to accomplish mission operations.



Systems View

The *Systems View* describes and interrelates the existing or postulated **technologies**, **systems**, and other resources intended to support the operational requirements.

Technical View



The *Technical View* describes the profile of rules, **standards**, and **conventions** governing systems implementation and **forecasts** their future direction.



DoD Architectural Framework (DoDAF) Products

Operational View (OV) →

1. High Level Operational Graphic
2. Operational Node Connectivity Des.
3. Operational Information Exchange Matrix
4. Organizational Relationship Chart
5. Operational Activity Model
6. Operational Rules
7. Operational State Transition
8. Operational Event / Trace
9. Logical Data Model

Technical View (TV)

1. Technical standards profile
2. Technical Standards Forecast

Systems (SV)

1. Systems Interface Description
2. Systems Communications Description
3. Systems – Systems Matrix
4. Systems Functionality
5. Operational Activity to System Traceability Matrix
6. System data Exchange Matrix
7. Systems Performance Parameters
8. System Evolution Description
9. Systems Technology Forecast
10. Systems Rules Model
11. Systems State Transition Description
12. Systems Event Trace
13. Physical Data Model





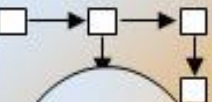
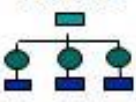
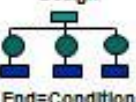




ALL (AV)

1. Overview and Summary
2. Integrated Dictionary

Spreadsheets, Text, Dynamic Models



DoDAF to Zachman Mapping

	Data <small>List of Things Important to Business</small>	Function <small>List of Processes</small>	Network <small>List of Locations Important to Business</small>	People <small>List of Organizations Important to Business</small>	Time <small>List of Events Significant to Business</small>	Motivation <small>List of Business Goals/Strategies</small>
Planner's View	Integrated Dictionary	Activity Model (List)	Operational Node Connectivity Description	Command Relationships Chart	Operational Event Trace	Capability Maturity Profile
Owner's View	<small>e.g., Entity Relationship Diagram</small> Logical Data Model	<small>e.g., Function Flow Diagram</small> Activity Model	Information Exchange Matrix	<small>Agent=Org Unit Work=Work Product</small> 	<small>Time= Business Event Cycle=Business Cycle</small> 	<small>End=Business Objectives Means=Business Strategy</small> 
Designer's View	<small>Entity=Data Entity Relationship= Data Relationship</small> 	<small>Operational Activity to Sys. Function Matrix</small> System Functionality Description	<small>e.g., Distributed</small> System Interface Description (High Level)	<small>e.g., Human Interface</small> Activity Model	<small>e.g., Processing Structure</small> 	<small>e.g., Knowledge Architecture</small>  <small>End=Criterion Means=Option</small>
Builder's View	Physical Data Model	System Interface Description (Detailed)	System Interface Description (Detailed)	System Interface Description (Detailed)	Operational Event Trace Systems Event Trace	<small>e.g., Knowledge Design</small>  <small>End=Condition Means=Action</small>
Sub contractor's View	 <small>Ent=Fields Rel=Addresses</small>	<small>e.g., Program</small>  <small>Funct=Language Stmt Arg=Control Blocks</small>	System COMMS Description	An Aspect of Multiple Products	 <small>Time=Interrupt Cycle=Machine Cycle</small>	<small>e.g., Knowledge Definition</small>  <small>End=Subcondition Means= Step</small>

DoD Architecture Framework Products

Operational View

Systems View

Technical View
(rules not explicit in Zachman)

Example: Enterprise Arch Thread

- US -> Security of nation
- DOD -> Defend the interests of the nation
- Air Force -> Air Superiority
- F-22 -> War Theatre Dominance
- ECM -> Air Defense Suppression



Example: Crusader 155mm Self Propelled Howitzer

- The Crusader self-propelled howitzer was being developed for the US Army as a replacement for the Paladin and the US Army requirement was expected to be for over 800 vehicles. In May 2002, the Crusader program was officially terminated by the Department of Defense because it was not considered sufficiently mobile or precise for the evolving security needs of the 21st century.
- United Defense reduced the weight and size of the Crusader vehicle allowing two vehicles rather than one to be transported on a C-5 or C-17 aircraft.



Product/System Architecture

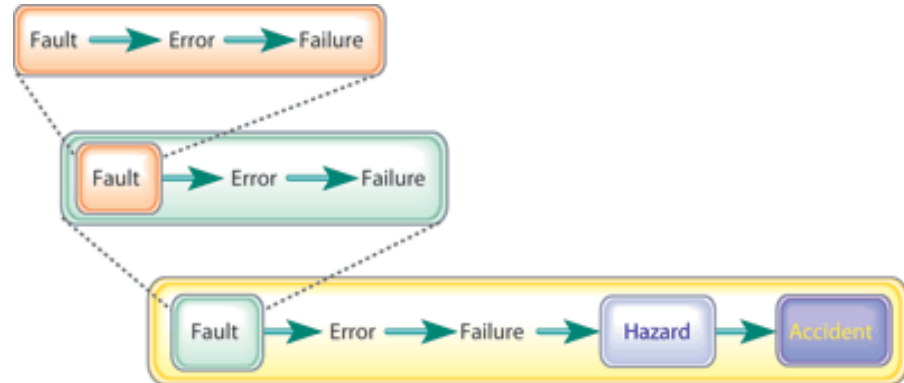
Patterns of safety architecture:

The main goal of the architecture patterns is to capitalize the expert's solutions in the field of safety. The re-use of these recurring and mature solutions will allow saving time in the systems architectures design and analysis phases.



Why be “safe”?

- Cascading effects can lead to life-threatening hazards
- High-Availability systems are normally not “safe”. They are designed to maximize “uptime”. The modes of failure are mostly “fail-stop”, with a take over by another system OR system restart.



Failure Condition Classification	System Development Assurance Level
Catastrophic	A
Hazardous / Severe Major	B
Major	C
Minor	D
No safety effect	E

Table 5-1, System Development Assurance Level Assignment



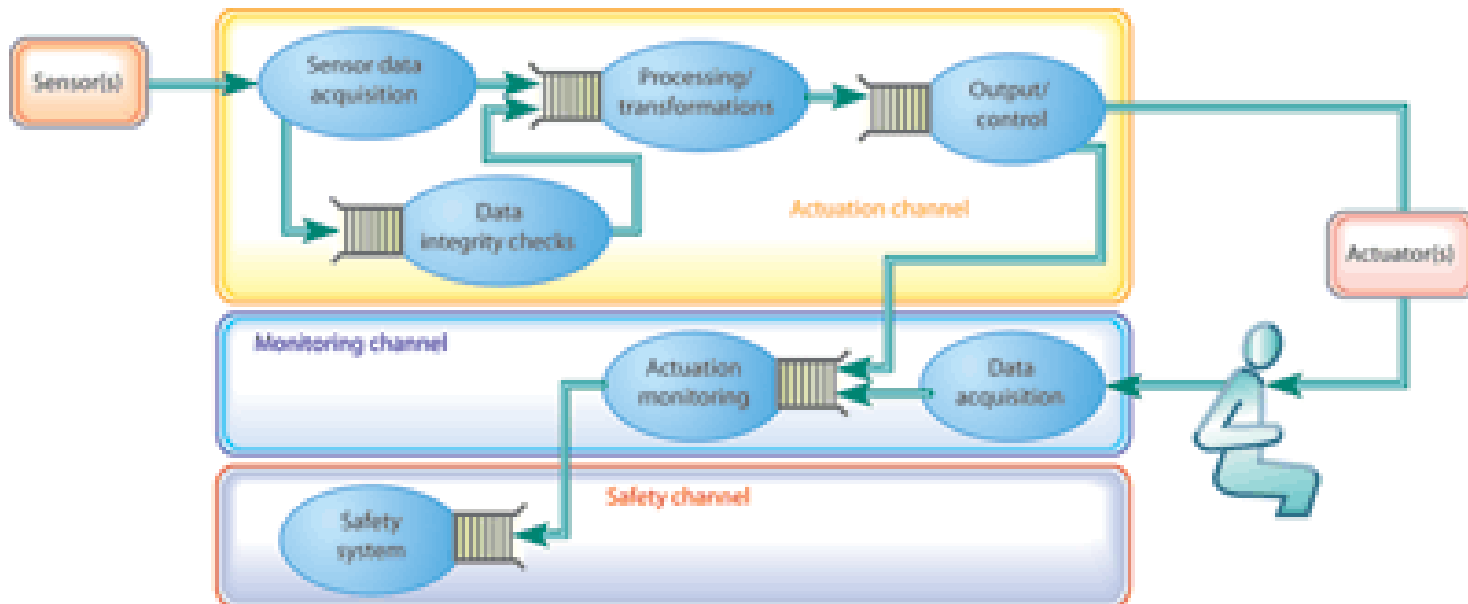
Product/System Architecture

- Type of Safety Architecture per ARP4754:
CERTIFICATION CONSIDERATIONS FOR
HIGHLY-INTEGRATED OR COMPLEX
AIRCRAFT SYSTEMS
 - Partitioned
 - Dissimilar Independent
 - Dissimilar
 - Active Monitor Parallel
 - Backup-Parallel



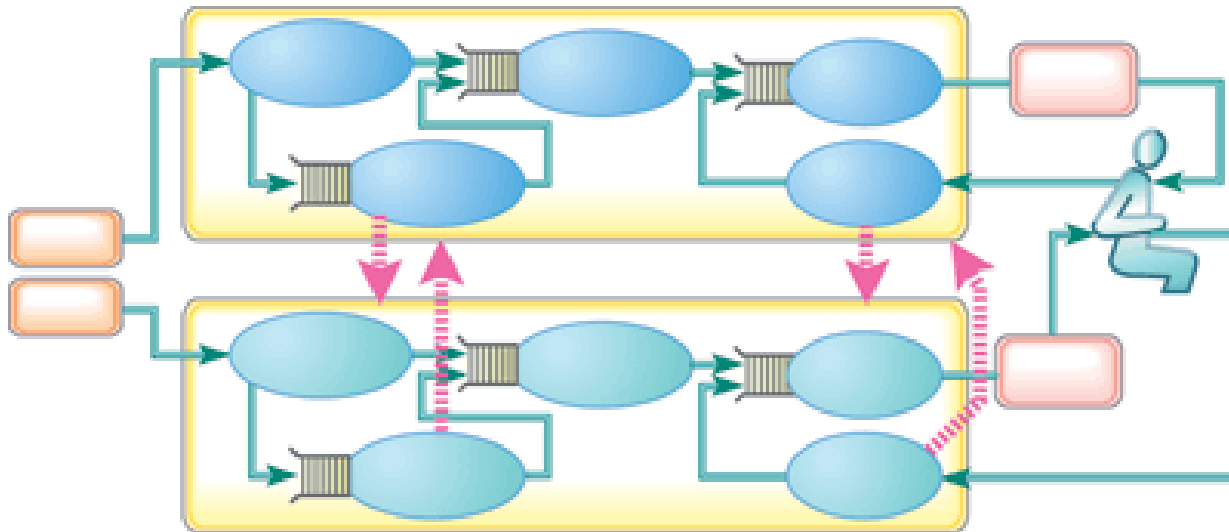
ARP4754 Architecture

- Active/Monitor Parallel Design
 - both portions are necessary to meet the integrity requirements.



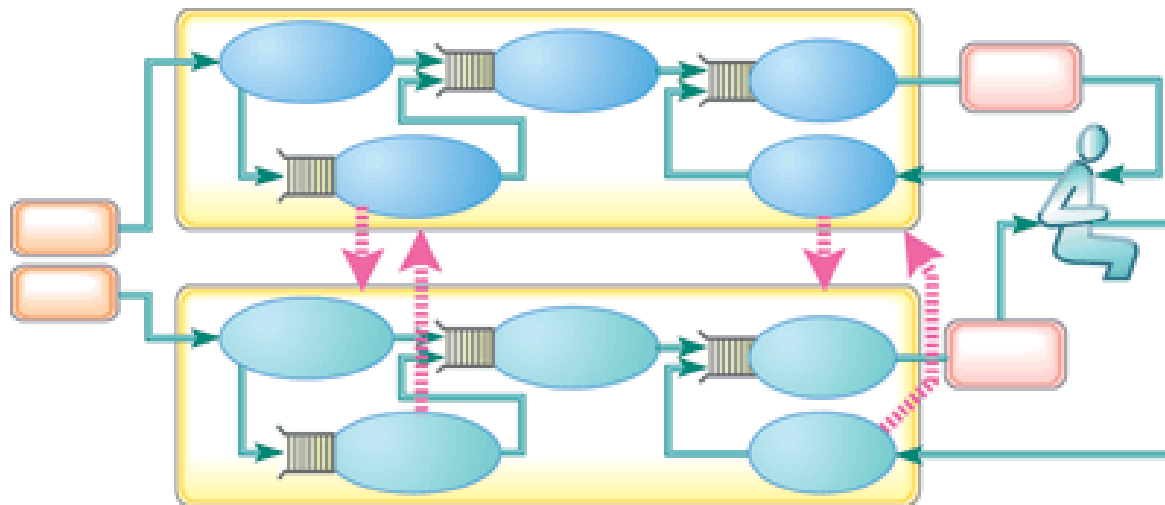
ARP4754 Architecture

- Backup Parallel Design (Dual Channel)
 - Backup only required to operate after the other system has failed.



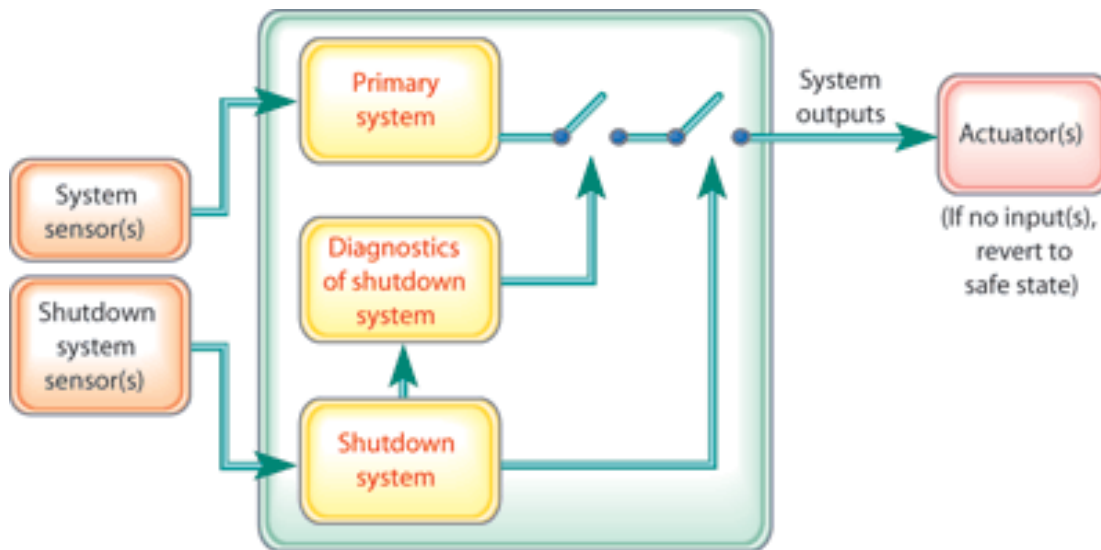
ARP4754 Architecture

- Dissimilar Independent Design
 - To be considered within this category, there must be substantial differences between the designs in terms of the means of preventing the top level failure condition
- Dissimilar Design
 - If multiple portions cannot be shown to be independent, then the primary portion should be developed to the development assurance level associated with the most severe failure condition



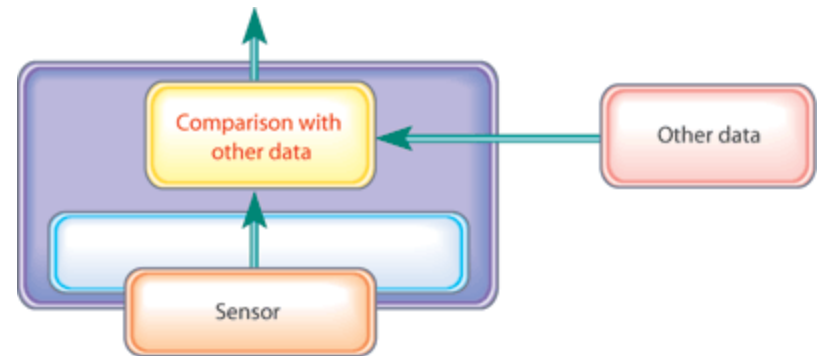
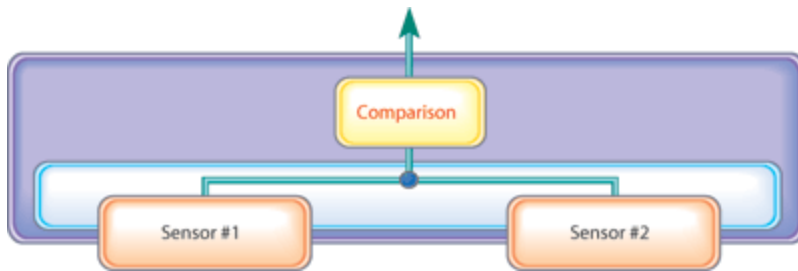
ARP4754 Architecture

- Partitioned Design
 - Partitioning is a design technique for providing isolation to contain and/or isolate faults



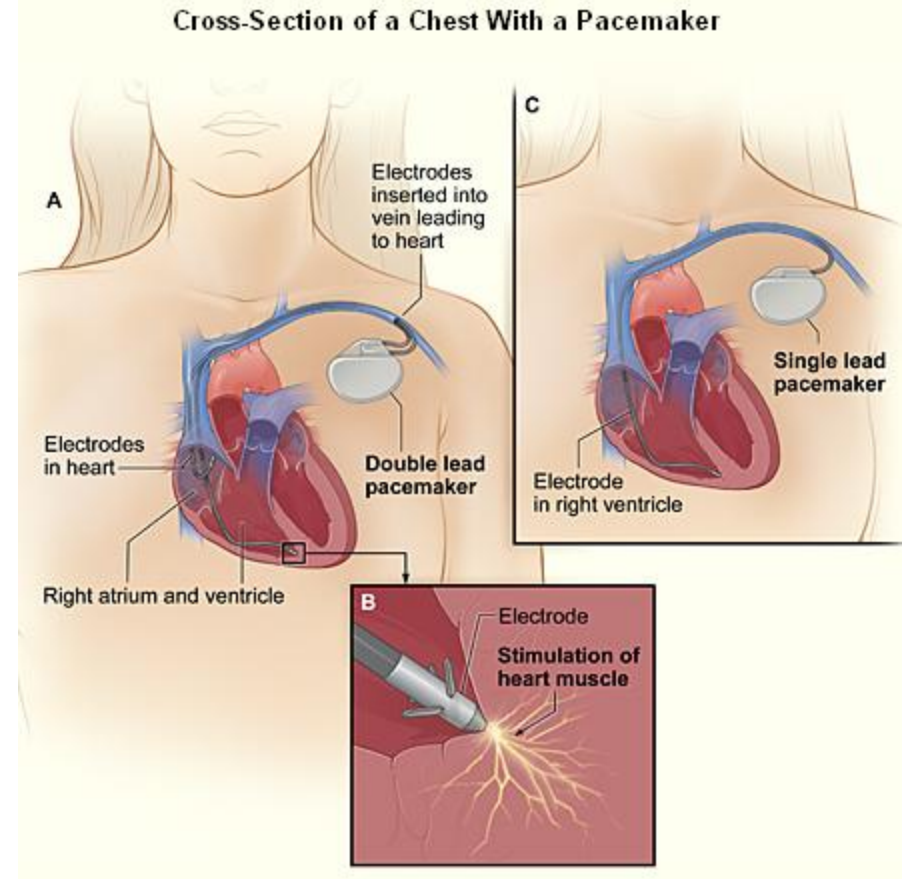
ARP4754 Architecture

- This can be applied to sub-systems or components.



Example: Pacemaker

- A pacemaker system consists of a battery, a computerized generator, and wires with sensors called electrodes on one end. The battery powers the generator, and both are surrounded by a thin metal box. The wires connect the generator to the heart.
- In order to minimize power drain, it is necessary to incorporate certain functions into the hardware outside the microprocessor.



Example: F-16 Stores

- For each program a unique interface would usually be implemented, usually also with a set of unique problems, such as the missile 'ghosting' problems experienced during the F-16 to AMRAAM integration (Ward 1993).
- In response to the problems of such an approach *MIL-STD-1760* an *Interface Standard for Aircraft to Store Electrical Interconnection System* was released by the US DoD to standardize aircraft/store interfaces.

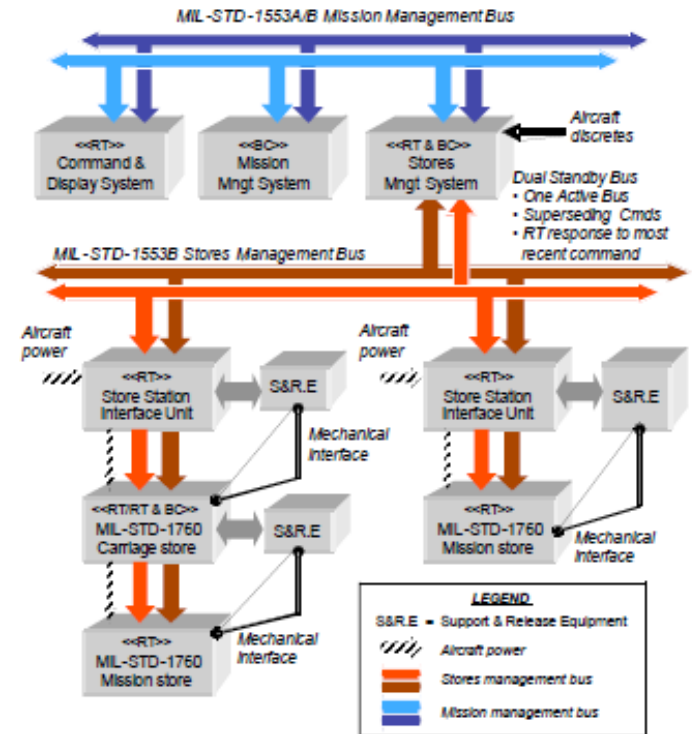


Figure 3-1 A distributed stores management architecture



Example: UNIX/POSIX

- UNIX/POSIX – abstraction of underlying hardware and OS services to provide a common API for cross-platform reuse. Layering of services to provide compounding effect.
- Unix-like operating systems are built from a collection of [libraries](#), [applications](#) and [developer tools](#), plus a kernel to allocate resources and talk to the hardware — [Hurd](#), [GNU's kernel](#) is actively developed, but is still some way from being ready for daily use, so GNU is often used with the kernel Linux.



Designing for Emergence: TCP/IP

- *“In the earliest days, this was a project I worked on with great passion because I wanted to solve the Defense Department's problem: it did not want proprietary networking and it didn't want to be confined to a single network technology.” - Vinton Cerf (Inventor of TCP/IP)*
- *“We had no idea that this would turn into a global and public infrastructure.” - Vinton Cerf*
- *“The Internet is based on a layered, end-to-end model that allows people at each level of the network to innovate free of any central control. By placing intelligence at the edges rather than control in the middle of the network, the Internet has created a platform for innovation.” - Vinton Cerf*



Example: CEV/Orion

“Chaos at the Heart of Orion”

Graveyard Spiral

Development: The 28 planning groups are meeting too late. The Constellation modules are already being designed without the guidance of any overall program plan. And so the spacecraft may turn out to be inconsistent with the plan that finally appears and incapable of supporting it.



Example: Aircraft Electric System

- Driving Requirements from ETOPS - Extended-Range Twin-Engine Operations Or “Engines turn or people swim”
- Aircraft use a variety of technologies and redundancy to provide electric power and its distribution,

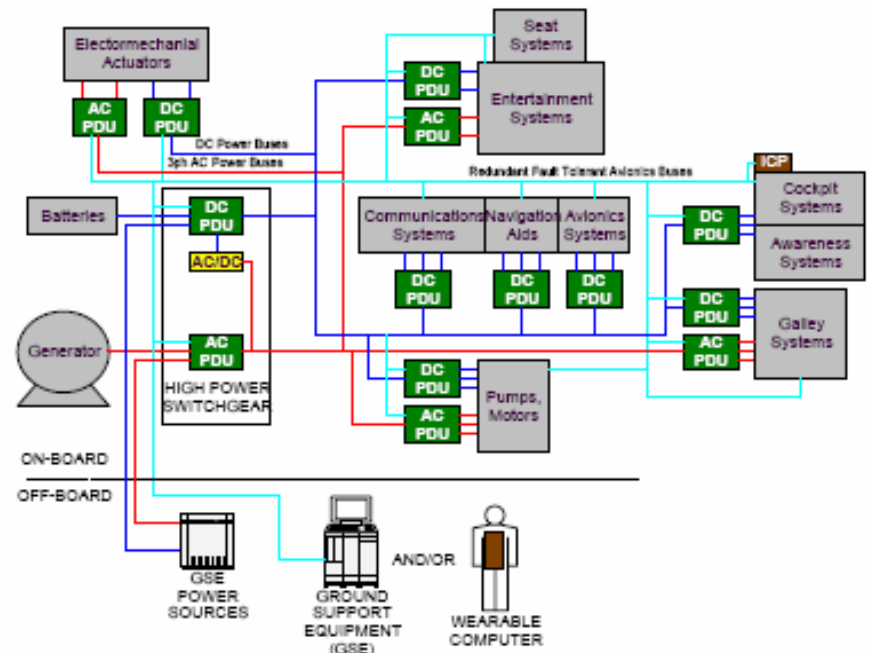


Figure 1. Generic Power Distribution and Management System Block Diagram

Lessons Learned: Arch at the Interfaces.

- Create a sound architecture
- Use abstraction
- Create structure that supports adding new capabilities easily.
- Humility is an asset
- Use interfaces to complete the architecture
- Be thorough about the details - Agreement & stability are more important than perfect performance
- Make it as bullet-proof as possible. – The more successful you are, the more stress you can expect to see at the interfaces over time.
- With interfaces, originality is not always a virtue
- Keep it as simple as practical...
- Be humble...



Proposal for Healthcare

- The current US healthcare system has been criticized many reasons recently.
 - The system lacks....
 - It is a chaotic system without common...
- However, Chaotic systems can:
 - Adapt to environment quickly
 - Spiral out of control with equal ease.



Russ's HCAF

- Create a Healthcare Arch framework so that capabilities of systems can be easily assessed and compared.
- But who owns it?
 - The hospitals? Doctors?
 - FDA? DOH?
 - A HC Czar?



Questions?

Thank you for listening to me while on my soapbox.



References

- Rehtin, E., “System Architecting”, Prentice Hall, Englewood, NJ, 1991.
- Advances in Computational Systems Architecting: Architecture Decision Graph Willard L. Simmons, PhD, INCOSE NE, January 14, 2009
- DoD Architectural Framework (DoDAF) - Michael Vickers, Martha Charles-Vickers, 1/4/2006
- Application of Patterns to Systems Engineering and Architecting - *INCOSE 2006 - 16th Annual International Symposium Proceedings*
- IEEE-Std-1471-2000: *Recommended Practice for Architectural Description of Software-Intensive Systems* Rich Hilliard, 11-14-2000
- Scott Workinger, Ph.D. SoS: Where’s the Beef? INCOSE Los Angeles Mini-Conference 2/7/09,
- <http://www.army-technology.com/projects/crusader/crusader1.html>
- Architecture of safety-critical systems By David Kalinsky, [Embedded Systems Design](#), (08/23/05, 06:08:00 PM EDT)
- The role of the computer in cardiac pacemaker technology. [Benedek ZM](#), [Furman S](#).
- 2002-01-3210 - A Configurable Solid State Power Management and Distribution System, John M. Maxwell, Jr., John H. Blumer and Blake Burden The Boeing Company, Phantom Works, Copyright © 2002 Society of Automotive Engineers, Inc.
- Wikipedia

