

Relationship-Oriented Architecture: A New Paradigm in System Engineering

Tobin Anthony, Ph.D.

Christopher Costello

Shawnee Heritage Government Solutions

July 10, 2007

Proprietary
Shawnee Heritage Government Solutions, LLC

Agenda

- Introduction
- The Problem
- Relationship-Oriented Architecture
- Example
- Case Studies

Introduction

- Tobin Anthony, Ph.D.
 - 12 yrs with NASA GSFC
 - 10 yrs in aerospace and defense systems
- Christopher Costello
 - 19 yrs in aerospace and defense systems
 - 12 yrs in small business engineering
- Shawnee Heritage Government Sol'n
 - Majority-owned by Shawnee Indian tribe
 - Native-American 8(a) exemption pending
 - Principals in business since 2003

ROA Concept

- Simple Is Better Than Complex
- A Picture Is Worth 1000 Words

Diagnosis: Chaos

The Problem: Inefficient management of design cycle leading to cost & schedule overrun

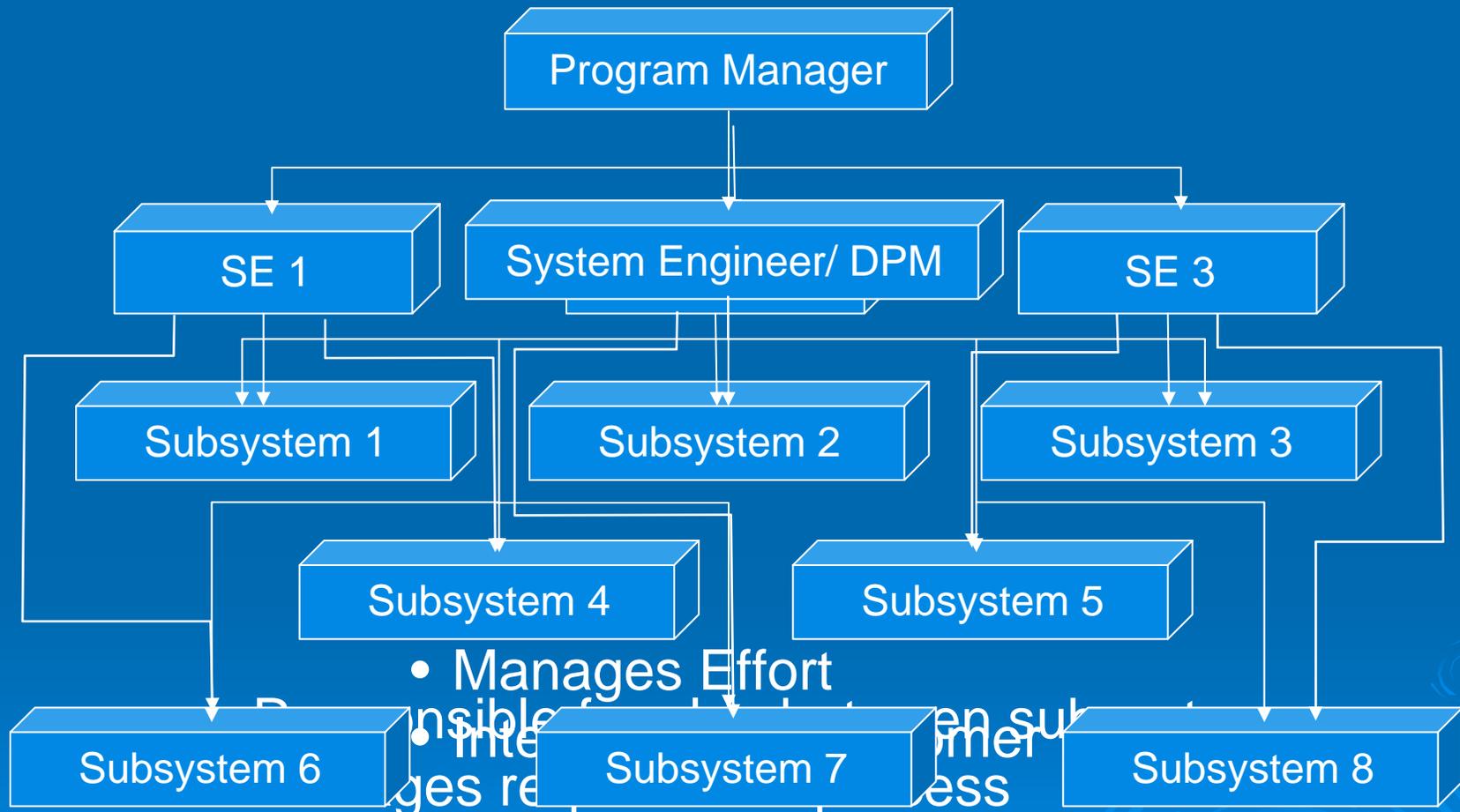
The Cure: Develop a system architecture to guide system design

But you end up with...

Micro Management



Typical Program Office



Only by using the SET tool can you set the entire system

ROA Solves

The tool expert by...

Being tool independent

The key man by...

Simplifying the process

Poor communication by...

Providing illustrative products

Micromanagement by...

Defining architectural relationships



What is ROA?

➤ History

- Developed by authors for use on UAV ground system program
- Honed by years of frustration seeing programs “winging it”

➤ Concept

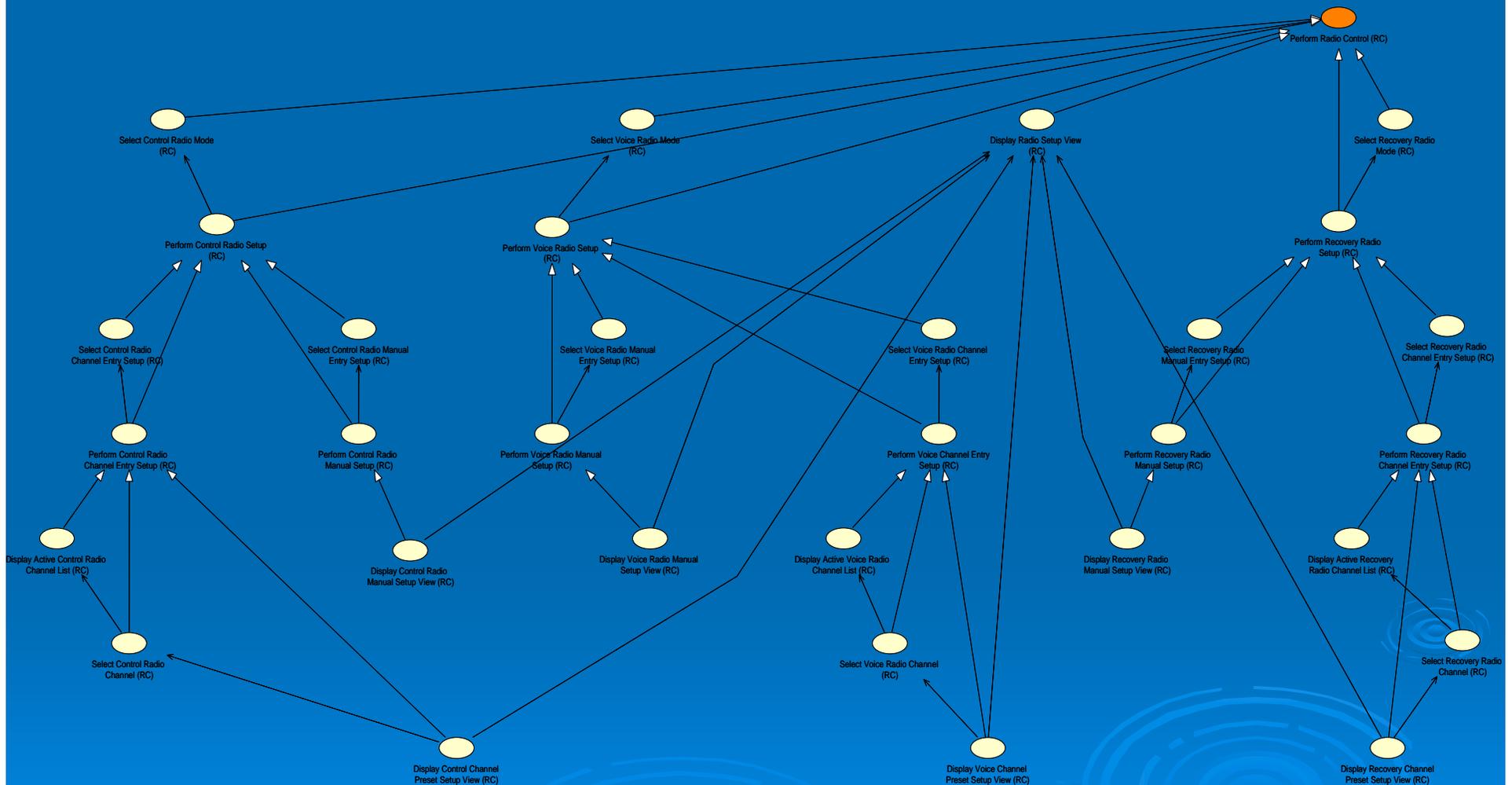
- Database, not tool-oriented
- Simple means of illustrating system by emphasizing relationships between elements

ROA Concept - Simplicity

- Rigorous, but not rigor mortis
 - Define a core set of system products
 - Illustration of the system architecture
 - Minimal set to document and communicate the design
- Engineer identifies systems elements and defines the relationship between them
- Examples of system elements are:
 - Requirements
 - Functions
 - Components (HW or SW)
 - Test procedures
 - Development milestones

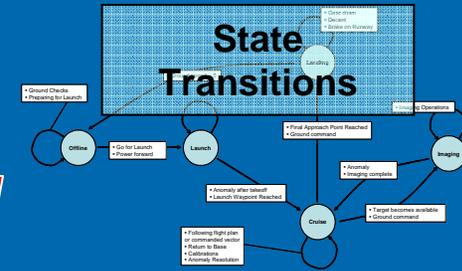
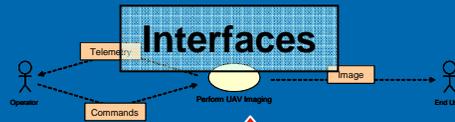
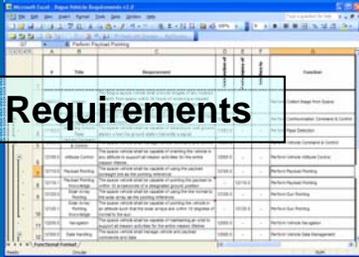
Relationships Between Elements Determine How the System Is Used & Built to Meet Requirements

ROA Concept – 1000 Words

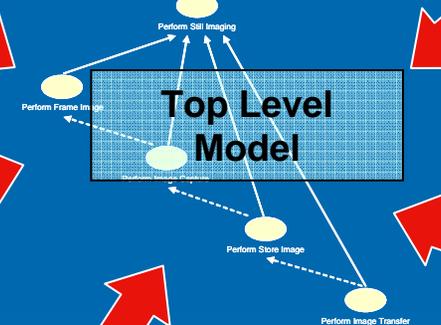


Glue the System Together

Requirements



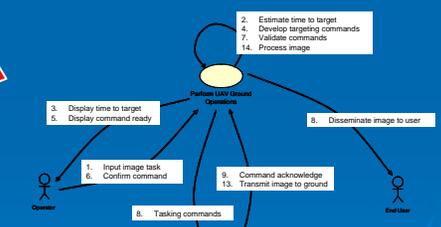
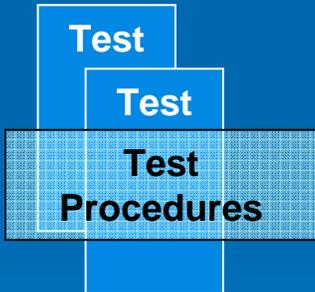
Top Level Model



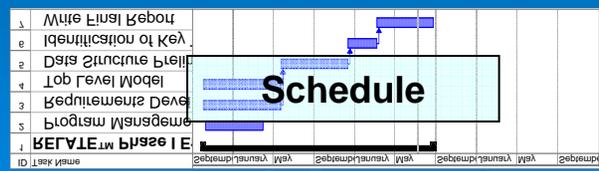
Test

Test

Test Procedures



Use Cases



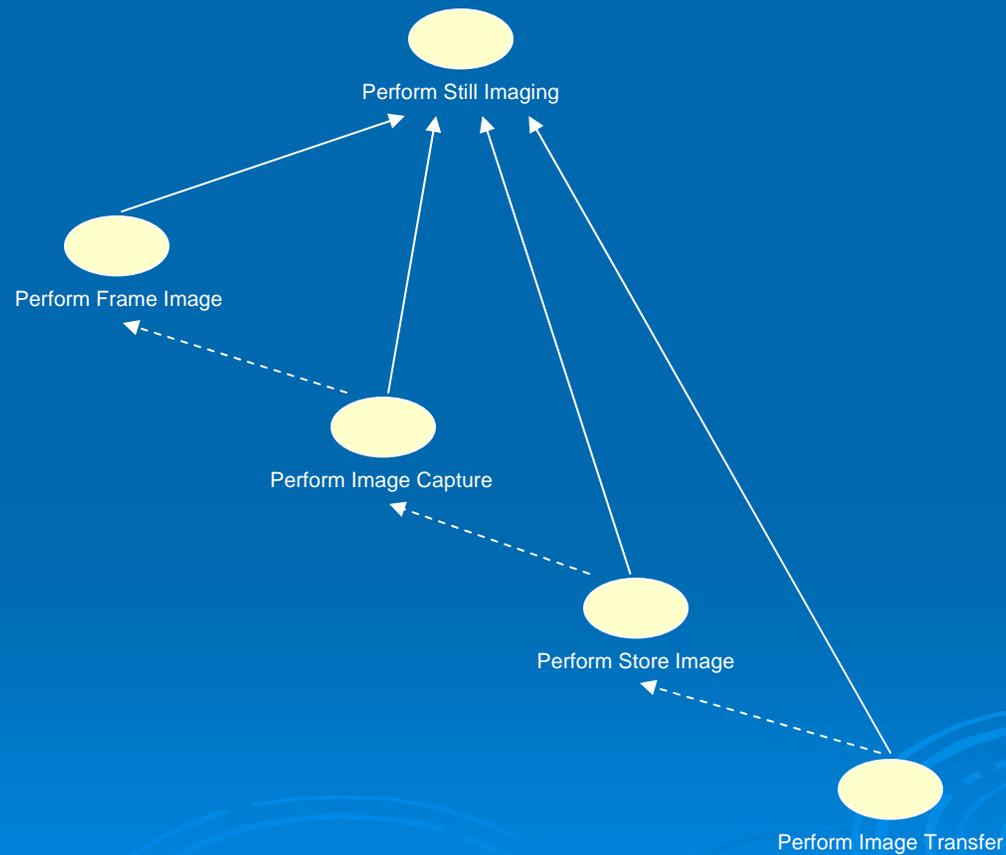
Schedule

Any Element



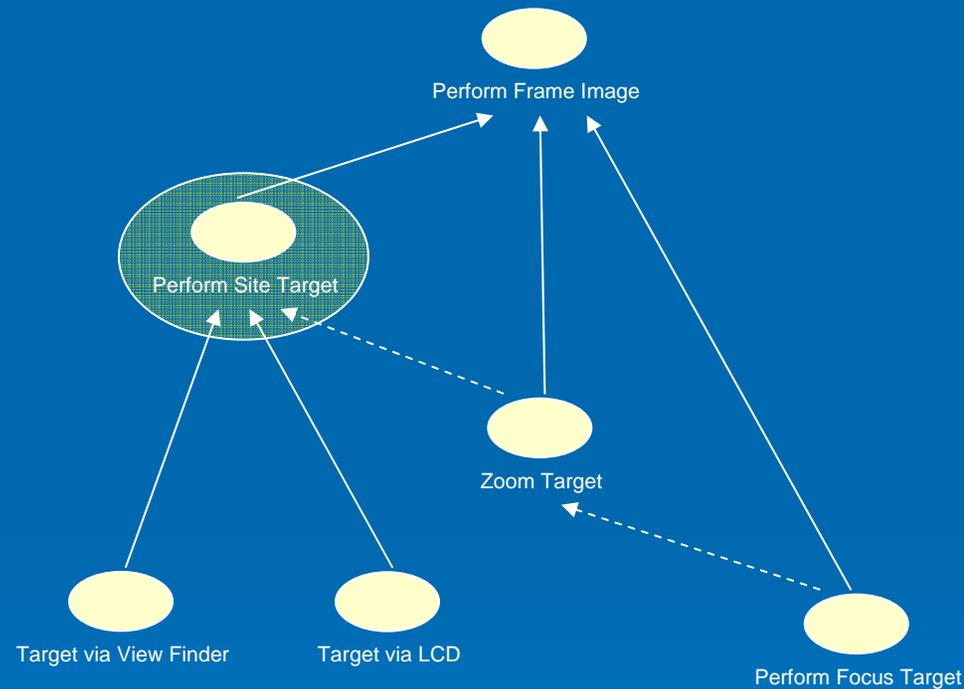
ROA: A Simple Example

System Requirement: *Take a Picture*



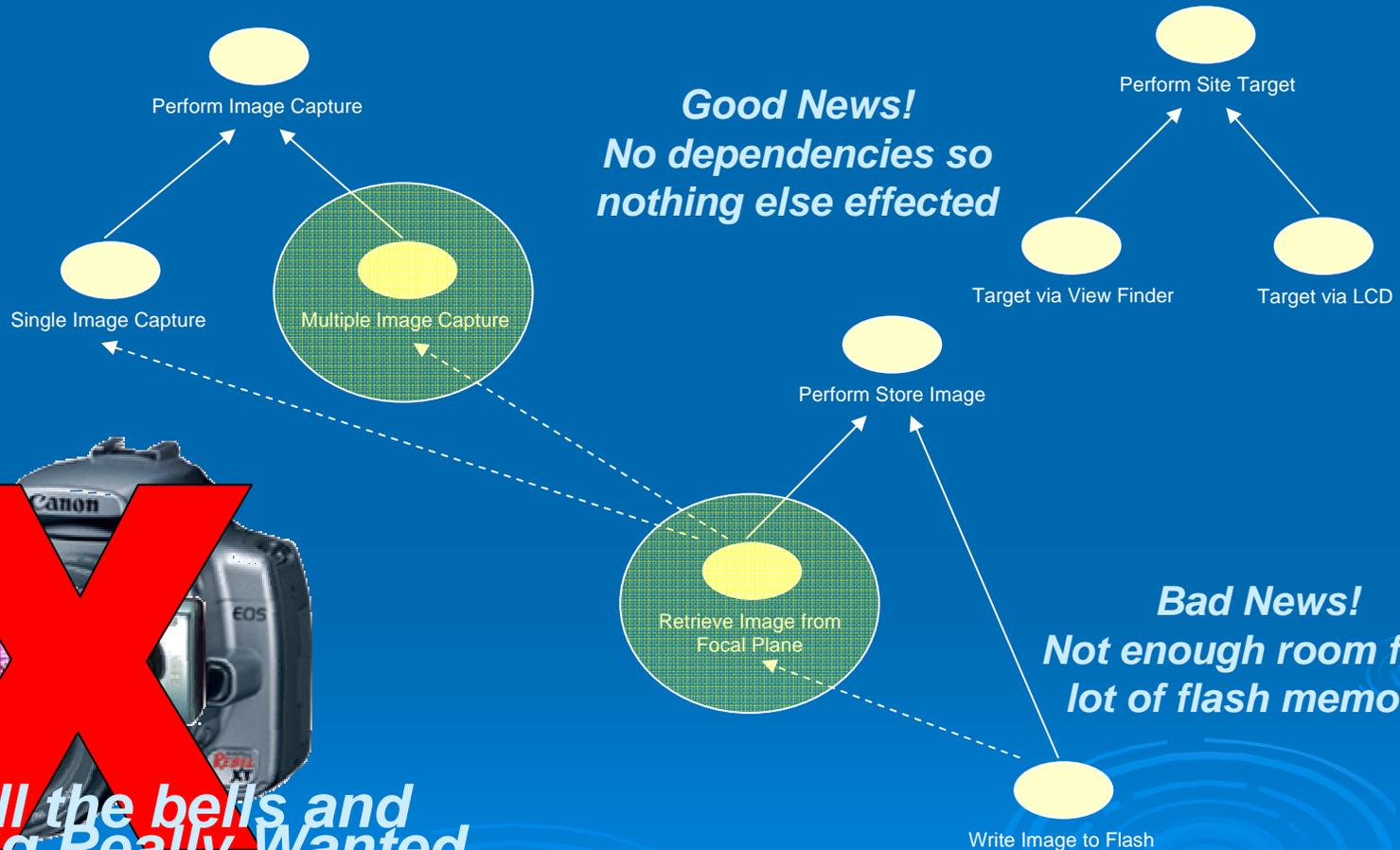
ROA: A Simple Example

System Requirement: *Frame Image* (derived)



ROA: Camera Development

System Requirements: Capture, Store Image, Site Target (derived)



Good News!
No dependencies so nothing else effected

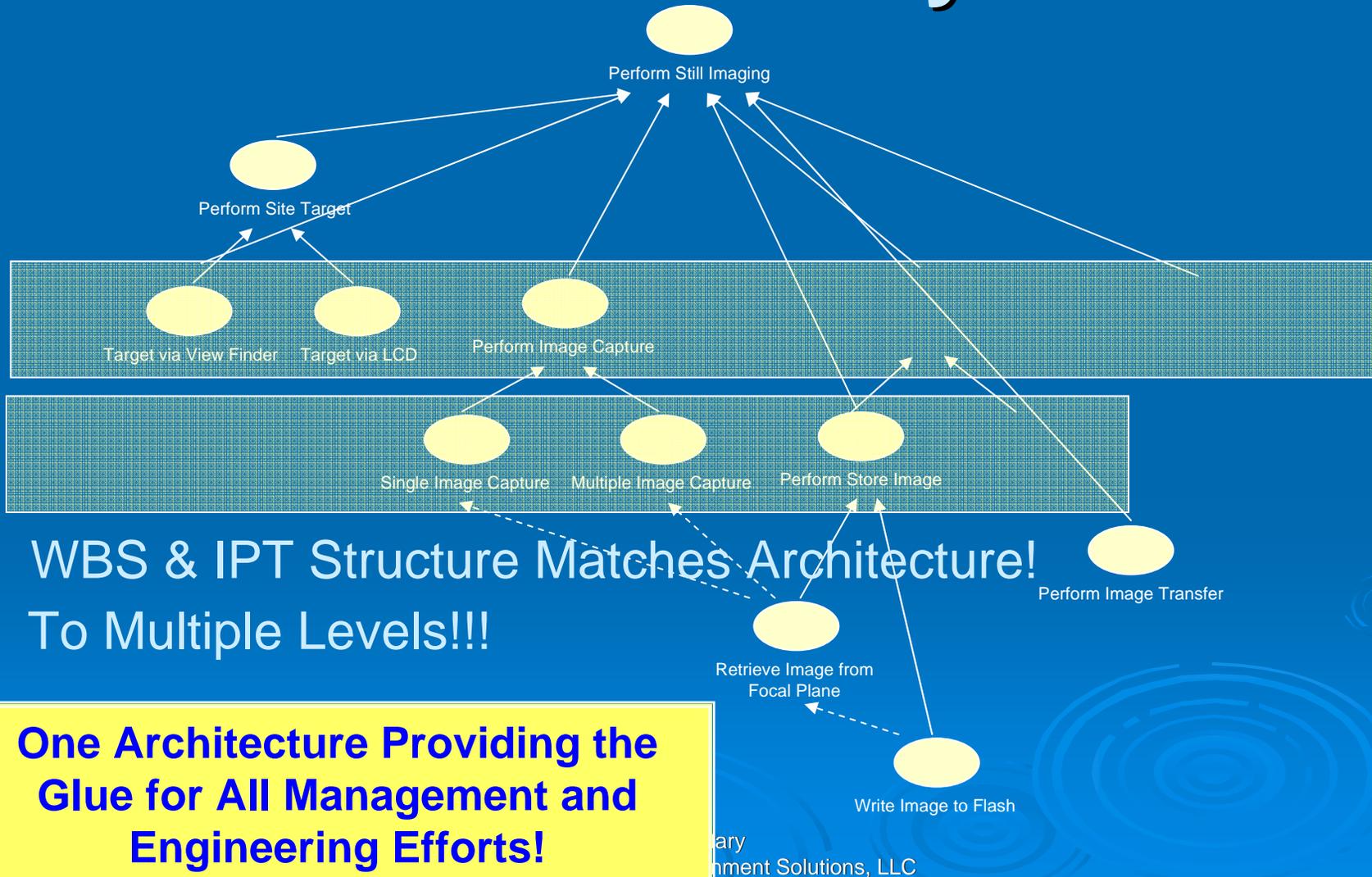
Bad News!
Not enough room for a lot of flash memory!



Not all the bells and Marketing Really Wanted Perfectly Engineered!!

ROA: TLM/Req/WBS

Consistency



Benefits of ROA

- Relationships between elements are defined
 - Enables evaluation of effects of changes to elements
- System design can be linked with management activities
 - WBS
 - IPT structure development
 - EVMS
- Elements are linked to requirements
 - Requirements without elements
 - Unverified requirements
 - Elements without requirements
 - Bloatware

Benefits of ROA

➤ Software Tool Independent

- Can develop architecture in any software environment
 - Core
 - DOORS
 - Requisite Pro
 - System Architect
 - MS Office

➤ Requires database back-end

- Provides link between individual elements and management artifacts
- Database can be simple spreadsheet

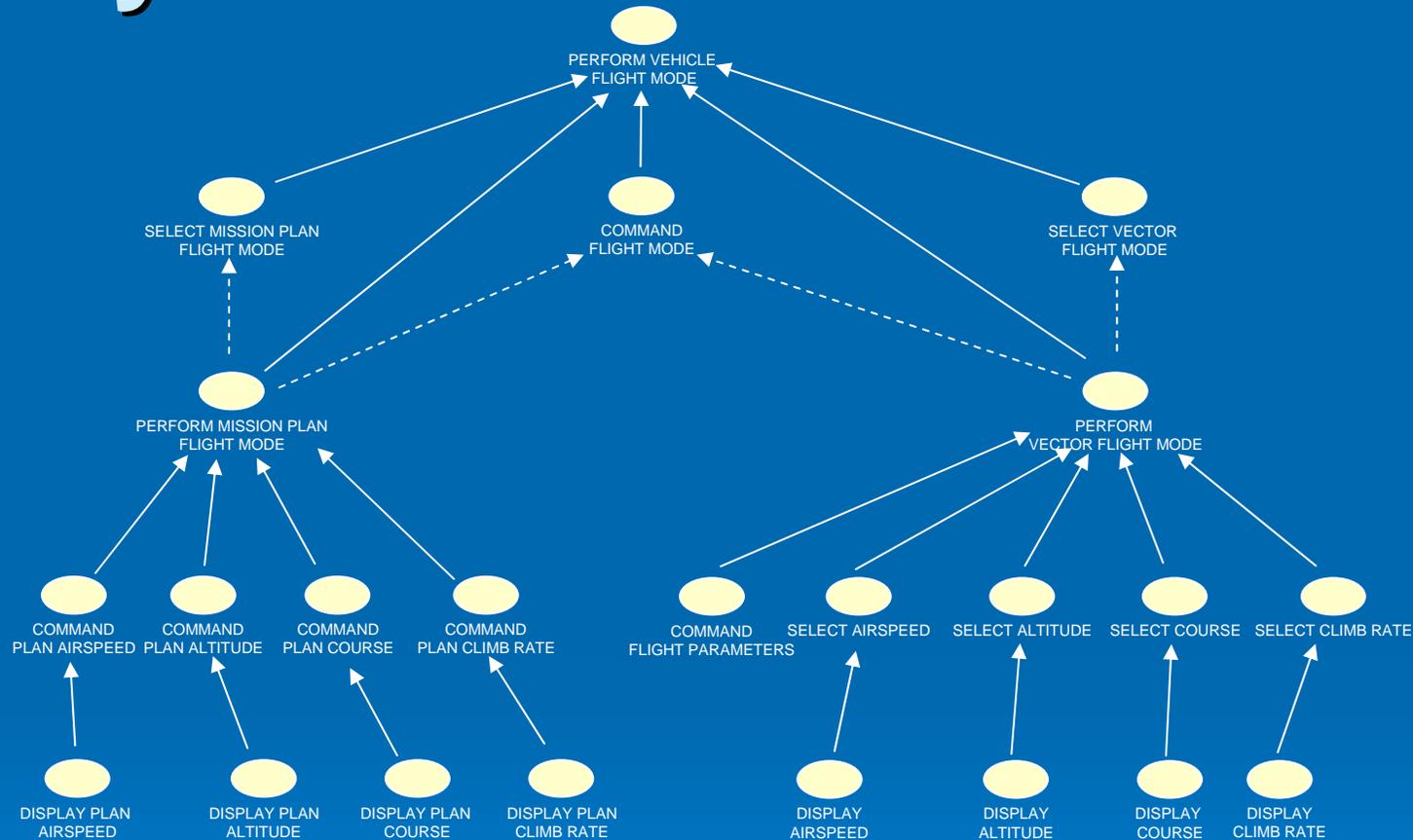
Case Studies

- Bringing UAV Ground System to CDR
- Forensic Analysis of Safehold Mode Software

Completing Design: Challenges

- Behind schedule
 - Engineering signed up to unrealistic schedules
 - Management did not understand engineering needs
- Management insight into progress limited
 - Mgmt to Engineering: “You’re Gold-Plating!”
 - Lack of trust through lack of communication
- Could not get to CDR
 - Customer and management did not agree on level of detail
 - System engineering not ready in customer opinion – poor CPAR

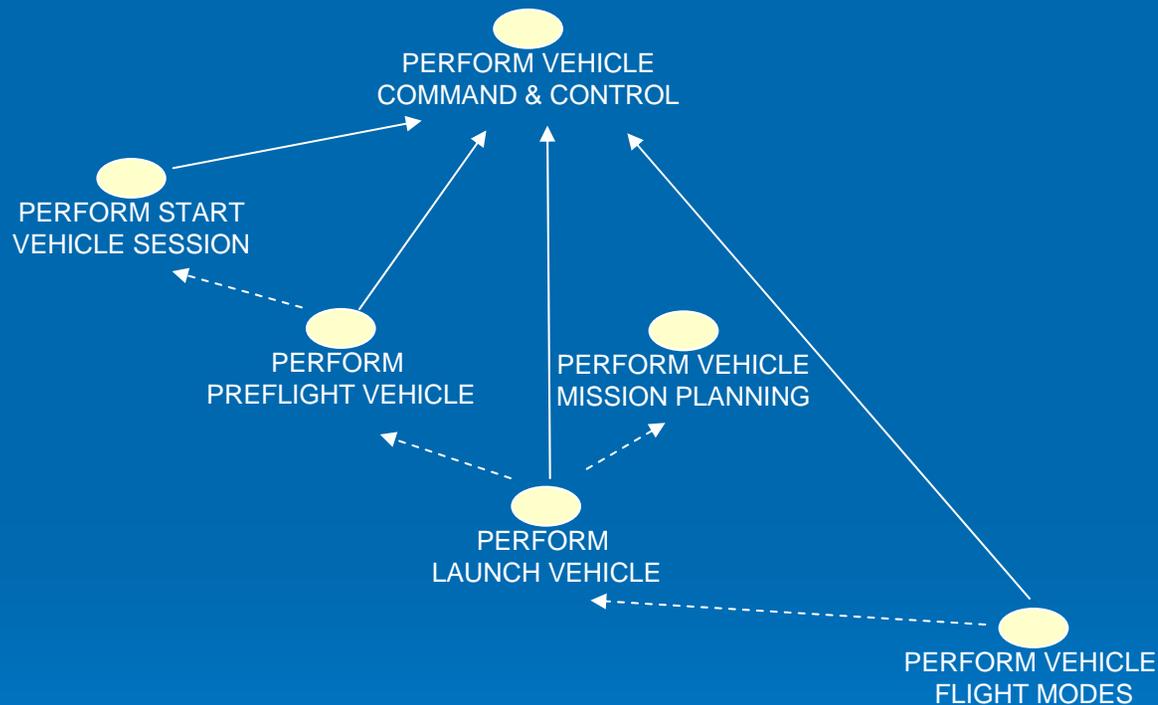
Why We Couldn't Get to CDR



➤ Not clear how this function fits into the entire system

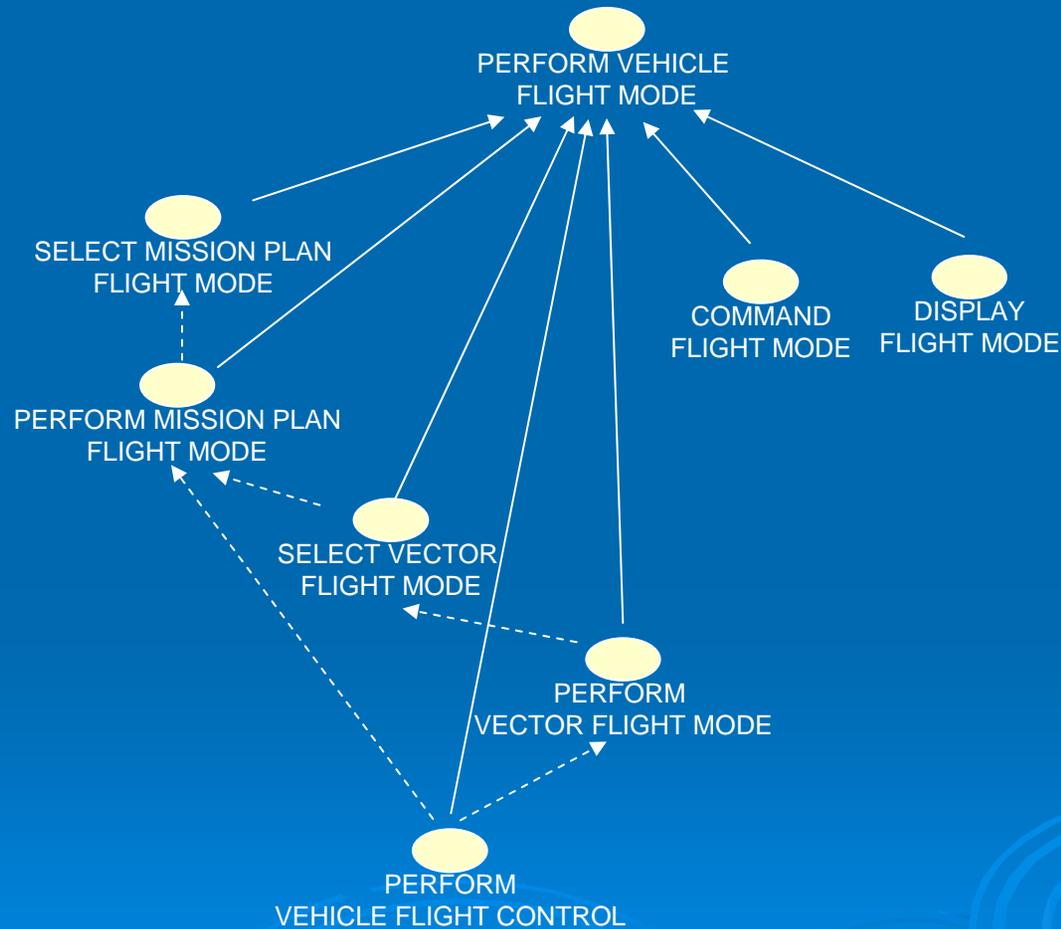
- Built separately for each mode
- Each model was implemented differently
- Very difficult to integrate!!!

Completing Design: Hierarchical Functional Decomposition

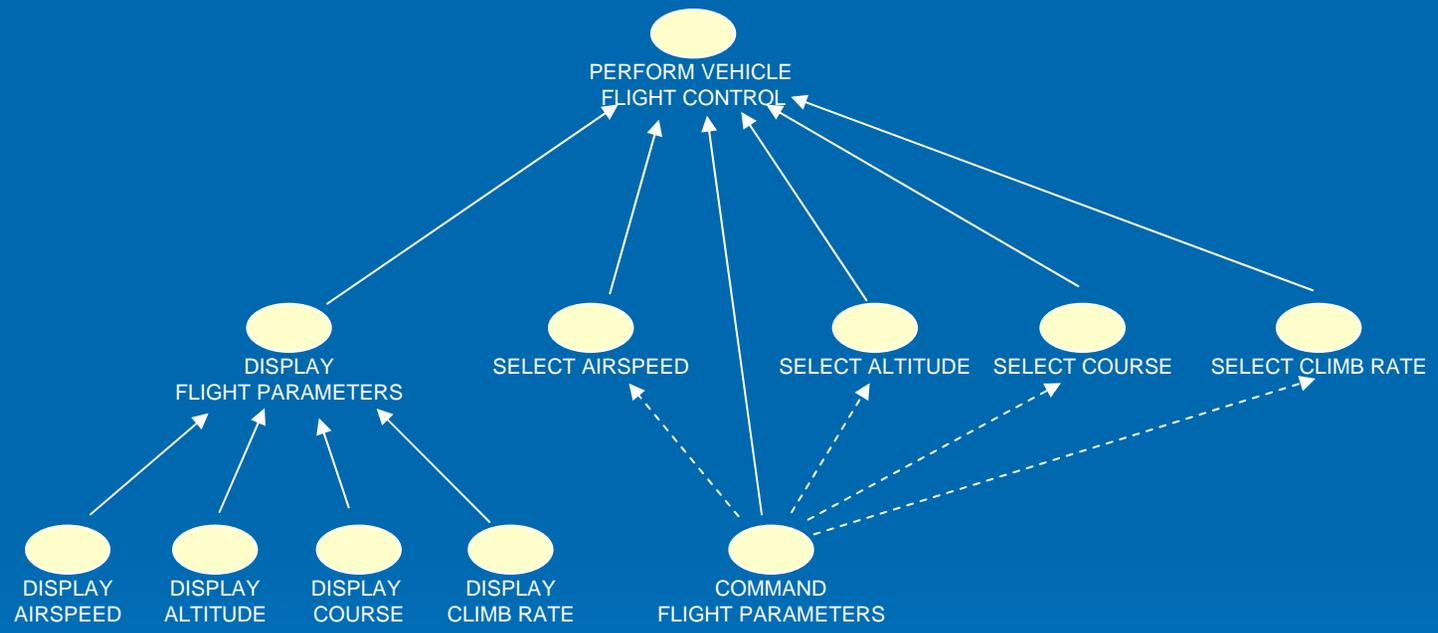


**Group Common Functionality And
Identify Dependencies**

Completing Design: Consolidate Common Functionality



Completing Design: Detail-level Use Case Definition



Only ~~Each~~ ~~flight~~ ~~mode~~ ~~variable~~ ~~can~~ ~~be~~ ~~linked~~ ~~to~~ ~~the~~ ~~software~~ ~~multiple~~ ~~flight~~ ~~modes~~

Completing the Design: ROA Solution

- Used the minimum set of artifacts
 - Requirements - what we have to do
 - Top-level model (glue) – how we are going to do it
 - Interface diagram – who we need to interface to
 - State diagram – when things are getting done
 - Sequence diagram – how the system behaves
- ROA Design was implemented in MS Office framework
 - Used Word, PowerPoint, Excel, and Access
 - Files were under CM
 - Accessible by management, engineering and customer

Everyone Working to the Same Goal, Focused on Objectives

Completing Design: Results

- Artifacts became management tool
 - Earned-value was related to each capability in TLM
 - Mgmt supported engineers when they had insight to progress
 - Customer gained confidence as they had insight to design
- Successful CDR
 - Customer engineers defended our design to their management
 - Highest CPAR rating
- Reduced build and test schedule because architecture was well-conceived and communicated!

**Next Phase of Program was Bid and Won Using
Architecture Artifacts to Show Enhancements!**

Forensic Engineering: Challenges

- Approached customer on ROA use
 - Customer liked methodology but wanted to see it applied to real problem
- Safehold Software flagged as concern
 - Contractor identified software robustness as potential issue
 - Concerned about algorithm “sinkholes”
 - Recommended full-scale redevelopment
 - Software had no documented design
 - Algorithm document existed
 - No knowledge of functional dependencies
- Large Red Team assigned to study problem

Forensic Engineering: ROA Solution

- ROA solution effort was two-pronged
 - Build top-level model (TLM)
 - Link functions to requirements
 - Done as a parallel effort to Red team
- TLM identified functional interfaces as the software was built
- ROA team scoured requirements and linked TLM elements to relevant requirements

Developed an Illustration of the Software Design

Forensic Engineering: Results

- No need to redesign software
 - Some areas of improvement identified
- Identified six requirements not being met by design
- Red team came to same conclusion
 - Relied on software and hardware-in-the-loop testing
 - Thorough, time-consuming, but not very broad
 - Delivered collection of opinions – no artifacts

ROA: 1.5 FTEs, 6 Weeks, Design Artifacts

Red Team: ~15 FTEs, 4 Months, Opinion

Future

- Develop ROA cookbook
 - Provide a standards-based means of publishing methodology
- Software tool
 - Develop MS Office-based toolset allowing ROA development
- Develop interfaces to external systems
 - Software development
 - Earned Value Management System
 - Toolsets to accompany established requirements software

Summary

- ROA enables the system engineer to become the system designer
- ROA provides a means of documenting system design
- ROA models lead to good implementation