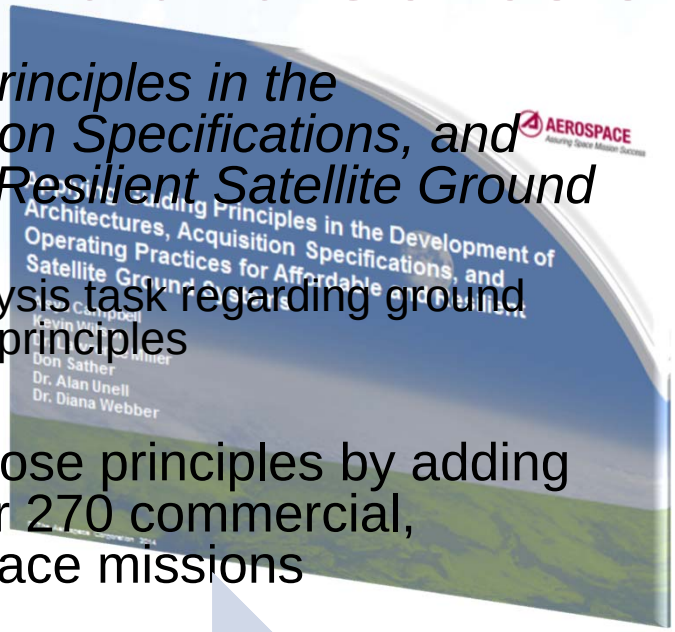
Guiding Principles for Successful Ground Systems – An Integrators Perspective

Gerry Simon
David Conway
Kratos Solutions Group

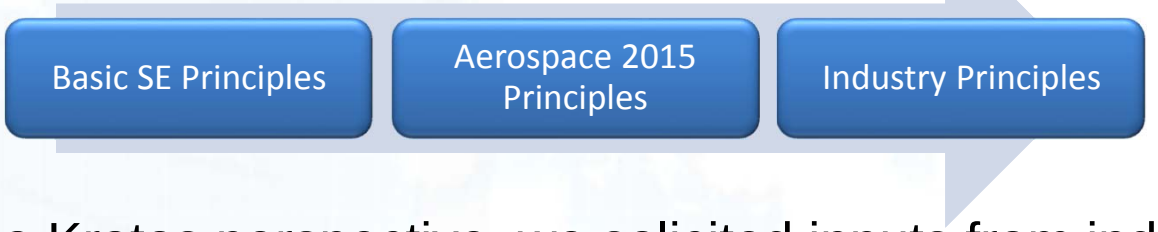
KRATOS

Background and Genesis

- In the 2015 GSAW, *Applying Guiding Principles in the Development of Architectures, Acquisition Specifications, and Operating Practices for Affordable and Resilient Satellite Ground Systems* - Campbell et al.
  - summarized the results of a significant analysis task regarding ground system development as a set of 12 guiding principles



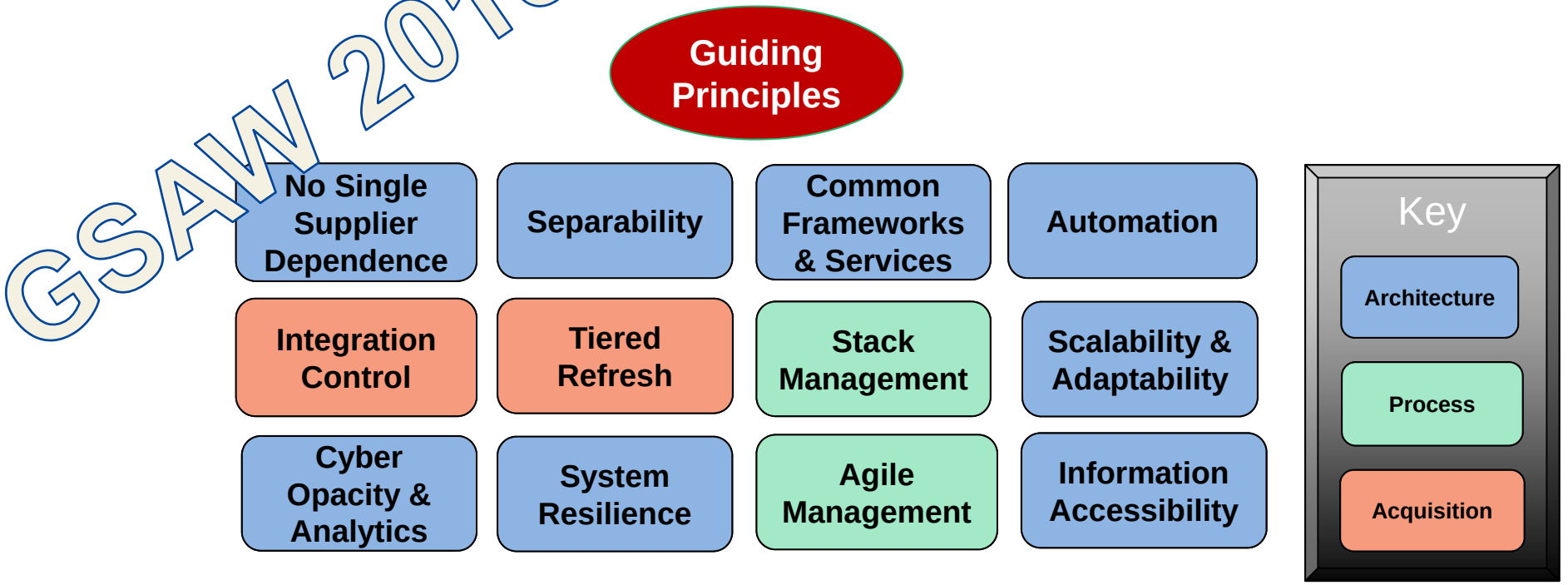
- This presentation *mostly* builds upon those principles by adding experience with our involvement in over 270 commercial, international, civil, DoD and national space missions



- Not just a Kratos perspective, we solicited inputs from industry peers big and small:

BOEING · LOCKHEED MARTIN · Orbital ATK · HARRIS · BRAXTON SCIENCE & TECHNOLOGY GROUP · AMERGINT TECHNOLOGIES · EXOSTRATEGIES

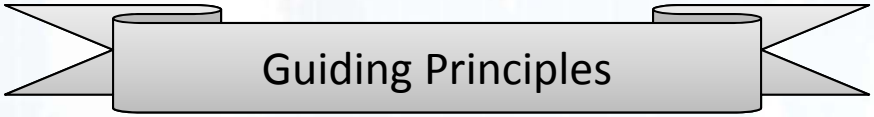
Guiding Principles for Affordable, Resilient Ground Systems



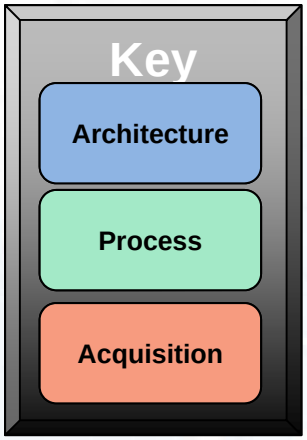
- **These twelve guiding principles have been generated from the findings to date and will be used to support viable EGS architectures and processes***
- **They may not represent a complete set, but they are the most dominant from the findings and reflect both architecture and acquisition guidelines**
- **Use of these Guiding Principles in the Architecting, Acquisition and Life Cycle Management of NSS Ground Enterprises will yield the most cost effective, resilient systems to meet mission requirements**

*Graphic and term taken from TOR-2015-00801. "Framework for an Affordable and Resilient Satellite Ground Enterprise for National Security Space Missions," The Aerospace Corporation

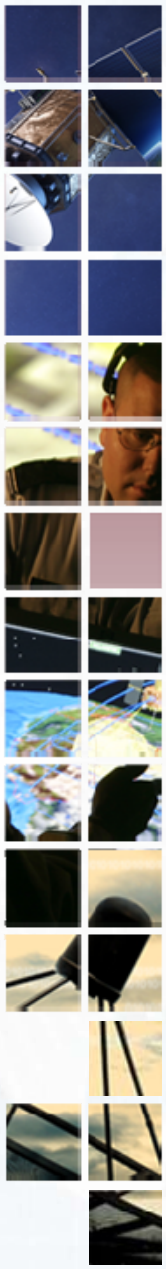
Guiding Principles for Successful Ground Systems



Open Interface Standards	Web Based User Interface	All Real-Time Interfaces Over the Bus	Drive Out Touch Labor
Define What is Not in the Baseline	Design For Change	Talent Over Process	Developer Acquirer Teamwork
Choose the Right Development Model	Running Code over PowerPoint	Free Isn't Free	Fear of Fixed Price is a Warning



- Top 4 Architectural, Process and Acquisition Principles – Integrators Perspective



Guiding Principle

Open Interface Standards

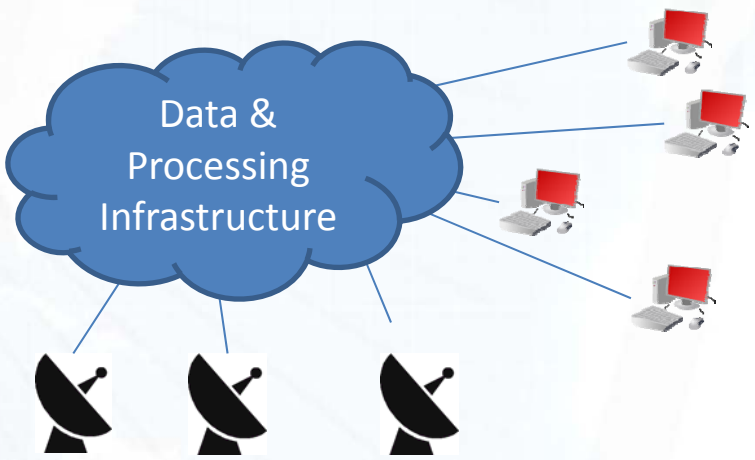
- Open Interface Standards drives how major system components are integrated into the solution
  - A standard that is publicly available has an open adoption/review process and has publically available rights to use
- Benefits
  - Required to support No Single Supplier Dependence principle
- Supporting Findings:
  - Closed interfaces impose significant friction for system upgrades, contractor changes and system evolution
  - TCP/IP & HTTP suite have transformed the world's economy
  - Well known Space Domain standards: TLE, CCSDS link, XTCE
- Counter Examples:
  - Open Standards (e.g. CCSDS MO Services) take years to develop and gain traction (e.g. IPv6)
  - Concern that open interface standards expose system vulnerabilities (security by obscurity)

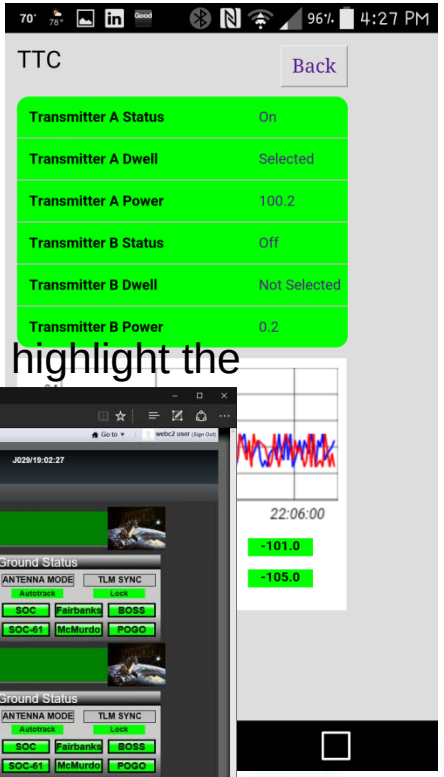
No Single Supplier Dependence

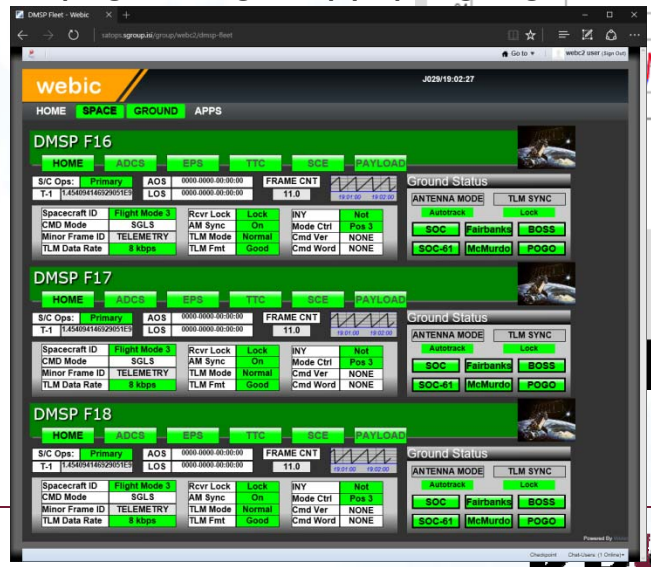
Qualities of Good Open Interface Standards
Backed by a Platform Independent Model - to facilitate migration to new technologies
"Be conservative in what you do, be liberal in what you accept" Jon Postel
Supported by an open standards development organization (SDO)

Guiding Principle
Web Based User Interface

- Web Based User Interfaces eliminate the need for any client footprint beyond a modern browser
  - HTML5 enables almost any robust user interface to be delivered in a secure way to widely dispersed clients
- Benefits
  - Greatly simplifies deployment and test
  - Shrinks cyber footprint
  - Enables core processing to be deployed to the cloud
  - OS independence: Windows, OS-X, IOS, Linux, Android
- Counter Examples:
  - Requires significant retooling of existing software base
  - Slow adoption of web based office tools (e.g. Google Apps) highlight the challenge in this conversion



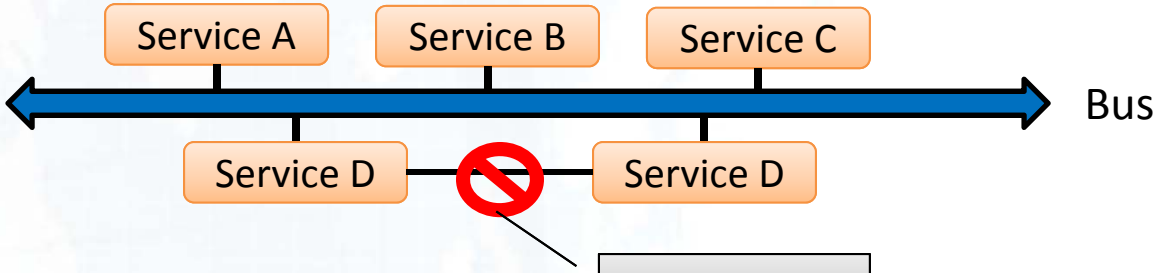




Guiding Principle

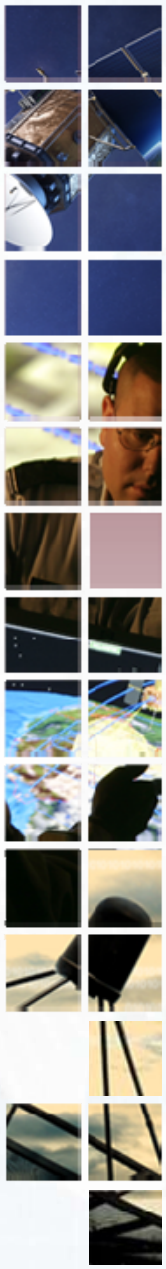
All Real-time Interfaces Over the Bus

- For Enterprise Bus architectures, the principle is to use the bus for all major component interfaces using open message standards. This includes state transitions and messages requiring authentication. No rogue interfaces – implement <u>all real-time interfaces over the bus</u>



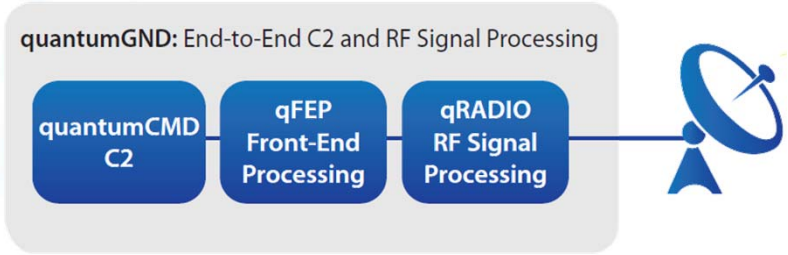
- Benefits
  - Enables loose coupling which enables Separabilty
  - Simplifies logging, monitoring and playback
  - Single cyber attack vector
- Supporting Findings:
  - GMSEC and various other implementations
- Counter Examples:
  - Requires significant retooling of existing software base
  - Beware the middleware upgrade quagmire
  - Possible bottleneck for extremely high performant systems

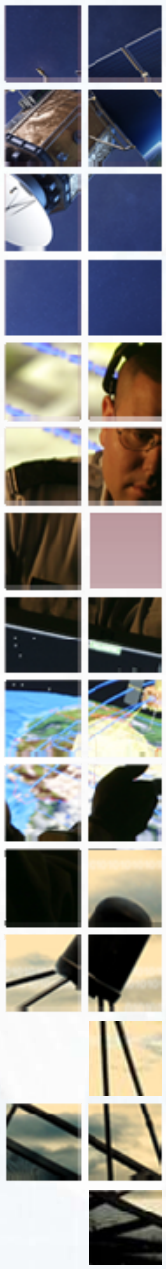
Separability



Guiding Principle
Drive out Touch Labor

- Dramatically reduce human in the loop during acquisition, development, deployment and operations in all phases by ruthlessly driving out touch labor
  - Must be a real quest – process entropy always forces touch labor up
  - Corollary: KISS
- Benefits
  - Quicker deployment
  - Labor is the most significant source of cost
  - Lower risk of error
  - Lower training/re-training cost
- Supporting Findings:
  - CCS-C Upgrade, quantumGND product line
- Counter Examples:
  - Risk of truly spectacular failure





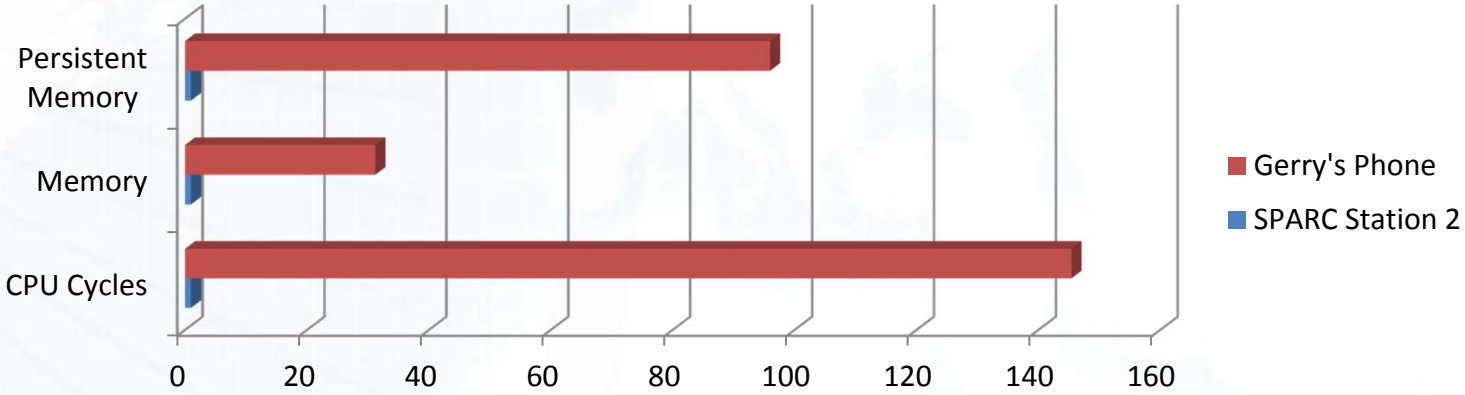
Guiding Principle

Define What is NOT in the Baseline

- Baseline management and configuration control are important but expensive. Based on the mission and to the level of mission maturity, define what is not in the baseline.
  - Recognize different levels of CM control even in operations,
    - Highest: Requirements, Architecture,
    - Medium: Configuration Files, software:
    - None: archive file content, monitors, position of the mouse
- Benefits
  - Minimal baseline control early in development shortens development cycles
    - Quicker development – all good projects start in a sandbox
  - Reduced cost during operations
  - System can more quickly adapt to mission requirement
- Supporting Findings:
  - The longer it takes to fix a problem, the more problem time you have
  - Agile*: the art of maximizing the amount of work not done*
- Counter Examples:
  - Eventually the lack of control will cause a problem and the response is always to add control (CM Entropy)

Guiding Principle
Design For Change

- Principle is to make system evolution an integral part of the ground system design. Over a ~15yr satellite lifetime, mission requirements will change. Ground system design is not static; design for change.
  - Cyber Security requirements are accelerating the pace of needed change
  - IT obsolescence forces change every 3-8 years
- Benefits
  - Maximize value of high dollar space asset investments
  - Relevancy
  - Lower lifecycle cost
  - Space assets are difficult to change so all adaptability must come from the ground system.
- Supporting Findings:
  - Moore's Law

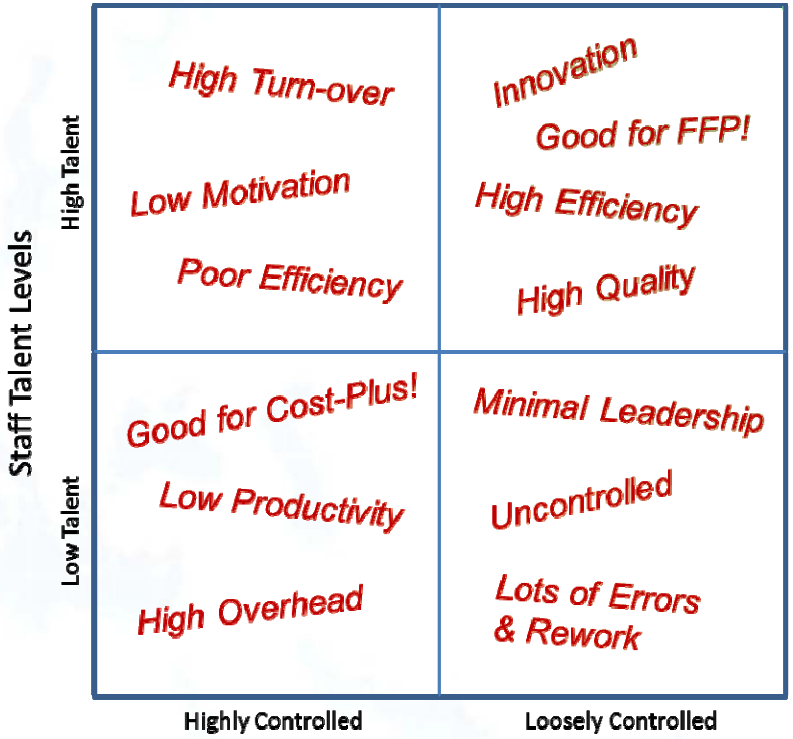


- Counter Examples:
  - Missions with short mission lifecycles

Guiding Principle
Talent Over Process

Agile Manifesto: *Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done*

- Focusing on Talent over Process
  - Faster development, higher quality, and less rework,
  - Creates an environment that fosters innovation and thrives on challenges
  - Better positioned for FFP
- Focusing on Process over Talent
  - Seeks the least common denominator
  - Requires significant cost & oversight
  - Slows progress and in the end does not guarantee results
- Counter Examples:
  - For talented people it's more difficult to curb expectations and control



Highly controlled environments tend to drive talent away

Highly talented people tend to seek other highly talented people

Only define process that are absolutely required

Guiding Principle

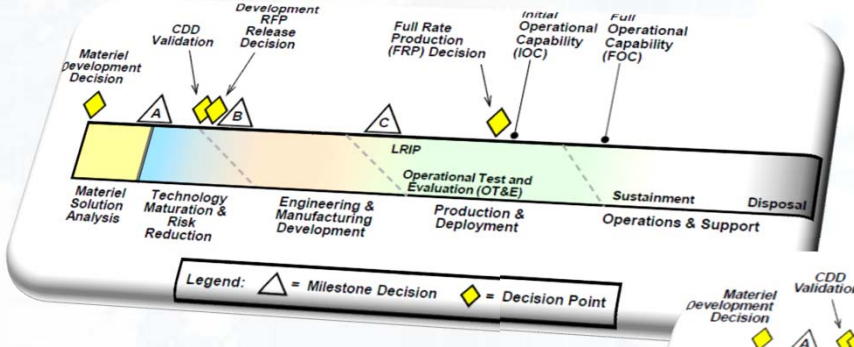
Developer/Acquirer Teamwork

- In successful ground systems deployments, <u>developer and acquirer team</u> to work cooperatively to a unified goal.
  - There will be changes in the ground system requirements, and the acquirer and integrator must work together to ensure that the ground system is delivered without exploding costs and drawn out schedule
  - Many of the attributes of "agile development" have really been around since the '70s, participation of the customer/end user in development & integration, prioritized requirements/desirements, frequent delivery and customer feedback.
- Danger Signs
  - The developer gets a “we know what we’re doing, leave us alone” attitude
  - The acquirer turns their watchdog role into an adversary role

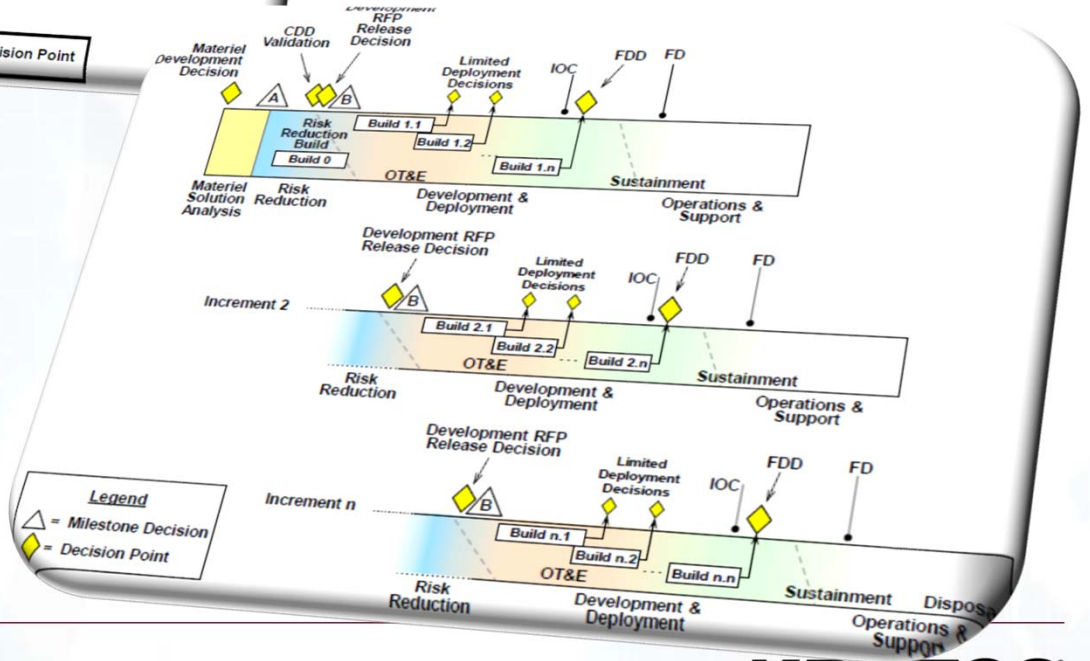
Guiding Principle

Choose the Right Development Model

- Waterfall, Spiral, Incremental, etc.
  - By necessity, space assets use a Hardware Intensive Model (Waterfall or V model). Tendancy is to default to the same model for the ground system. Choose the right development model.

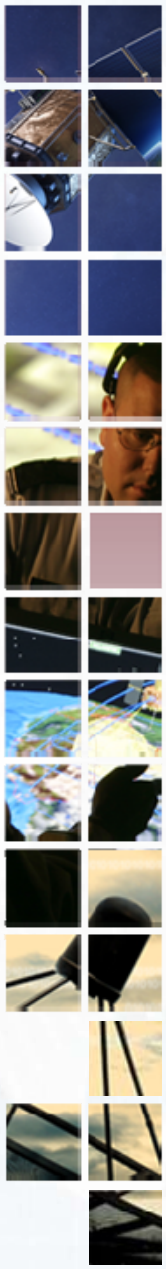


DoDI 5000.02
Hardware Intensive
Program



Incrementally
Deployed Software
Intensive Program

KRATOS



Guiding Principle

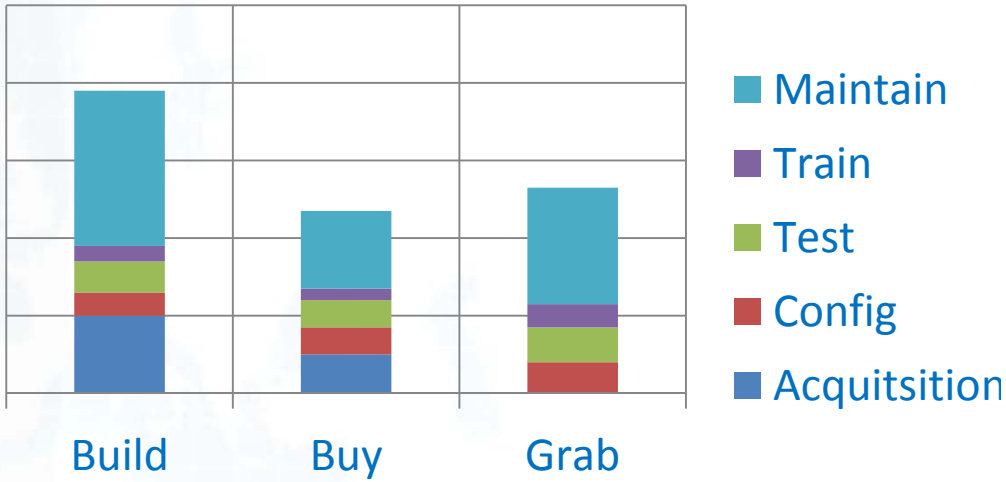
Running Code over PowerPoint

- Whenever possible, bias acquisitions based on ‘eyes on’ running software vice glossy brochures, slick websites or PowerPoint. Running code over PowerPoint.
  - Good marketers are seldom good developers Corollary: Good developers are seldom good marketers
- Supporting Findings:
  - Agile Manifesto: *Working software is the primary measure of progress*.
  - The CCS-C acquisition featured a ‘demo’ phase with TT&C on an actual spacecraft
- Counter Examples:
  - Draws out decision times and adds cost to the acquisition

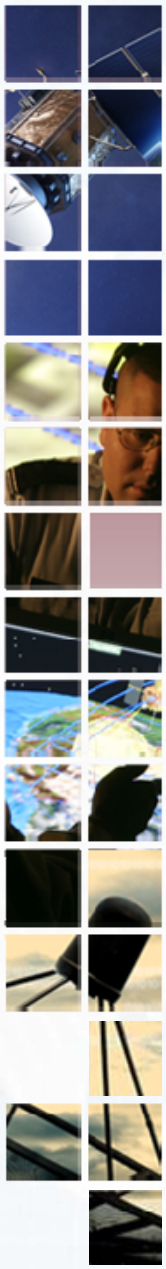
Guiding Principle

Free Software Isn't Free

- Free & Open Source Software (FOSS) and Government Off the Shelf Software (GOTS) have justifiably made big inroads into satellite operations centers
  - However, all software must be installed, configured, trained, integrated, and maintained even free software isn't free.
  - Not all open source licenses are equal
- Software Lifecycle Cost:
  - Build vs Buy vs Grab



- Example: qGnd full ground system for the price of ~3 labor months
- Counter Examples:
  - Linux, Firefox, JBoss, MySQL, Git, Docker, Bugzilla, Eclipse, Maven
  - BOSSIE (Best Open Source Software) software



Guiding Principle

Fear of Fixed Price is a Warning

- Supplier hesitancy to accept a firm fixed price contract, indicates one (or both) of two things:
  1. We aren't confident in our ability to deliver
  2. We aren't confident in the acquisition

Either way, <u>fear of fixed price is a warning</u>

- Benefits
  - FFP price predictability
  - Puts some restraints on 'extras'
  - Reduced accounting burden for both parties
- Counter Examples:
  - Requirements do change and sometimes fixed price really isn't reasonable
  - Fear of unknown unknowns and external dependencies


Questions?