GSAW Breakout Session

Process Mismatch with COTS-Based Systems: Problems and Solutions

Developing and Negotiating Requirements

Donald Sanders,

John Brown

Integral Systems, Inc.

April 2, 2004

Integral System Background

- Integral Systems, Inc. (ISI), provides satellite ground systems
  - Founded in 1982, 350 employees
  - Headquartered in MD; Offices in CO, OH, & Toulouse
  - Three subsidiaries; RT Logic, Sat Corporation, Newpoint Technologies
- We produce COTS software packages, and offer COS software suites
  - First to market with commercial software for command and control
  - Largest installed base of command and control systems in the world
  - Over 120 command and control systems delivered successfully on five continents
  - 50% world market share for commercial satellites
- We act as system integrators for turnkey systems
  - We provide a suite of COTS solutions
  - We enhance our products through bundling with 3rd party hardware and software
- Prime Contractor for CCS-C
  - Command and Control System – Consolidated
    - Monitors and Controls MILSATCOM Satellites

INTEGRAL SYSTEMS

Requirements Development (Commercial vs. Government Contracts)

- Initial requirement definition should be limited to high level only
  - Don't force a design, or accommodate "COTS looking for a solution"
  - Less is better

Commercial Contracts

- Typically requirements are defined within several pages of a "Statement of Work" type document
- Any "low level" requirements included are the exceptions to the rule and are usually only specified when something is not obvious
  - For instance, there is no need to specify that a design for new car has a steering wheel instead of a yoke unless you expect to use the car in an unconventional way

Government Contracts

- Requirements are specified more formally
  - Allows more a more equal playing field among competitors
  - Required to meet various FAR (Federal Acquisition Regulation) clauses
- Many times "low level" requirements are specified simply to meet a heritage "Ops Con"

INTEGRAL SYSTEMS

Requirements Development (Commercial vs. Government Contracts)

- Subsequent mapping of requirements into either COTS vs. custom code allows quick ID of product applicability
- Requirements tracking, mapping, and reporting maintains clear relationships and uncovers holes

Commercial Contracts

- Generally only high level mapping is done, very little documentation is produced to formally show the mapping
- Testing of delivered product can be informal, many times a comprehensive "demonstration" can be done to sign off on a system

Government Contracts

- Formal traceability is documented in various Product Specification and Requirements documents
- Government contracts tend to be "longer" with more builds
  - More sophisticated tracking is needed so that only the appropriate requirements are tested at each phase

INTEGRAL SYSTEMS

Requirements Development (Commercial vs. Government Contracts)

- COTS allows quick prototyping
  - May lead to new requirements, altered requirements, changed mapping
  - Can provide initial verification of architecture

Commercial Contracts

- Prototypes are crucial to helping our customers develop an Ops Con
  - Many times requirements are written AFTER a prototype has been demonstrated
- Changes to requirements are usually minimal due to the fact that requirements are at a high level
  - When things “change” it usually simply means that a refinement to an interpretation of a requirement has been made

Government Contracts

- Prototypes can be beneficial to demonstrate capabilities
- Lots of early customer interaction is encouraged but can impact the overall progress of the program
  - Contract Negotiations and Systems/Software Development follow two separate path that must meet
  - Good Systems Engineering must be in place to ensure that developers do not act solely on “comments” and that legitimate changes can progress swiftly through the contracting approval process

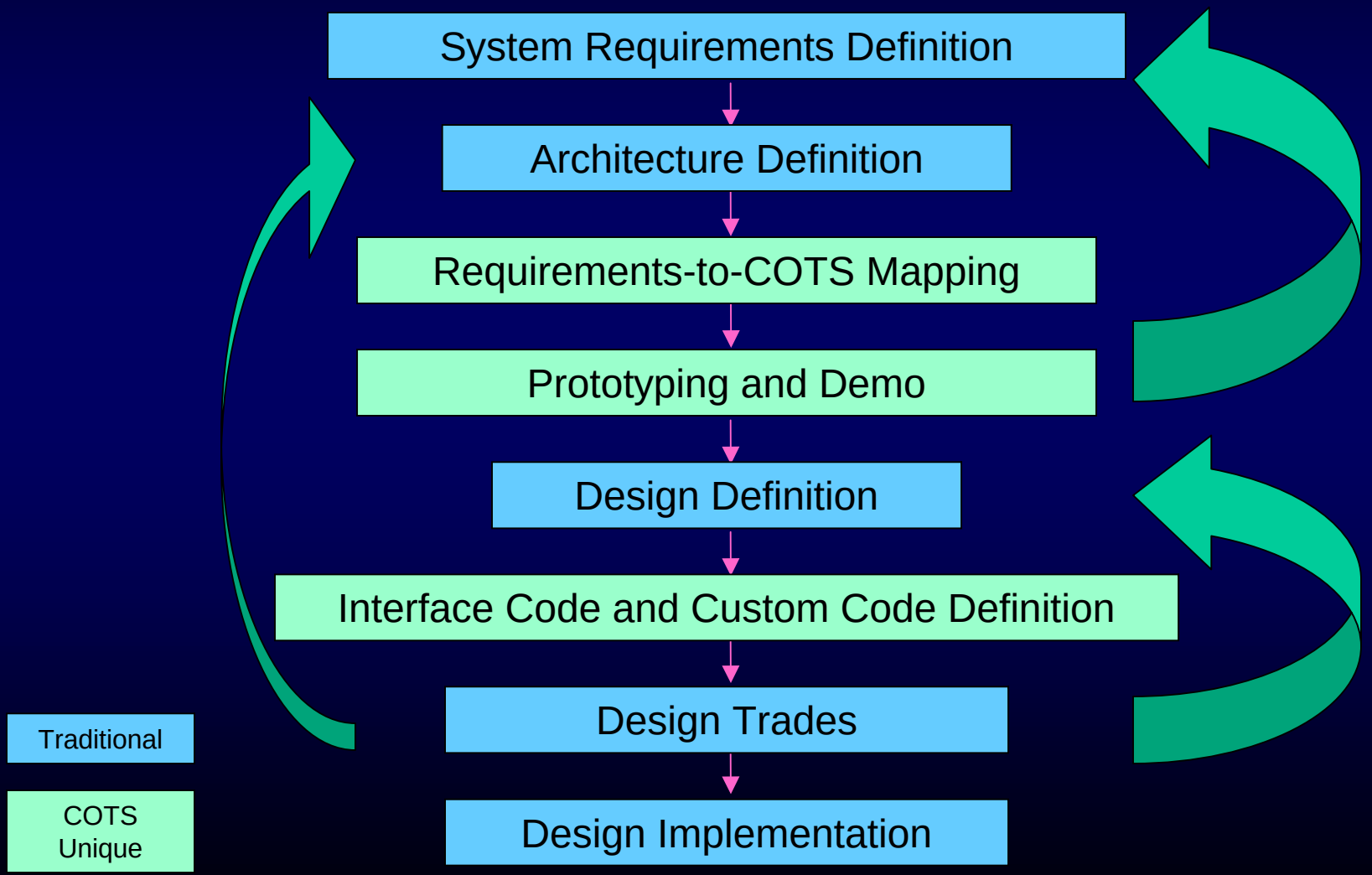
INTEGRAL SYSTEMS

Requirements Development for COTS-Based Systems

- COTS requirements should stop at functionality, not go into detailed design (simplifies testing too)
  - If a requirement specifies some type of design, then it may be impossible to meet the requirement with a COTS product even though the COTS product can meet the intent of the requirement
  - Writing a test procedure to verify a requirement is easier to do if the requirement is at a functional level
    - At the test procedure approval phase, if additional detail needs to be added to a procedure to verify something very specific, it can easily be accommodated
    - This allows the test to be focused on what is important
  - It may be difficult to modify a COTS product to include new requirements
    - Even though a company may have access to COTS source code, the product is generally maintained by a separate group whose staffing, funding, and schedule are driven by factors outside any single program

INTEGRAL SYSTEMS

Augmented Process



INTEGRAL SYSTEMS

CCS-C Example

- For CCS-C the requirements, and therefore architecture, cleanly mapped to COTS components
  - In Phase 1 design competition, requirements were defined, as well as a logical architecture, which included large COTS components
    - Requirements that were first overly restrictive were re-examined and improved
  - In Phase 2 award phase, architecture was refined to include over 30 different COTS components
  - Our COTS-based architecture facilitated a clearer understanding of the system, and how it would be built-out
  - Very few trades were necessary with customer requirements
- Systems Engineering played a key role
  - The DOORS DB tool was used for tracking/mapping requirements to COTS components, and custom elements and components
- CCS-C is an excellent example of a good contractor/customer working relationship
  - Partnership rather than antagonistic

INTEGRAL SYSTEMS

Requirements Development Conclusions

- Keep the requirements high level
  - Allows for easier functional mapping into COTS and custom code
- Allow for iterative process of definition, mapping and refinement
- Keep COTS requirements at black box functional level
- A shoulder-to-shoulder approach between customer and contractor is necessary
  - This approach provided flexibility, and accommodated demands of “process tracking” without being overtaken by process
  - Allows for more detailed requirements to be added under already existing requirements for clarification
- A strong systems engineering effort is necessary
  - Keeps everyone aware of requirements and how they are going to be interpreted and implemented
  - Provides a single focus point (lower level IPTs defined the details and presented them to the community through the SE IPT)
  - Maintained a traceability of requirement changes

INTEGRAL SYSTEMS

GSAW Breakout Session

Process Mismatch with COTS-Based Systems: Problems and Solutions

Creating and Validating Architectures

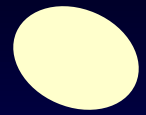
Donald Sanders,

John Brown

Integral Systems, Inc.

April 2, 2004

Which came first, the Architecture or the COTS?



- From the beginning…
  - Traditional approaches to requirements and architecture development was linear and most if not all systems were custom
  - Functional decomposition lead to subsystem definitions
  - COTS products were eventually developed that could meet specific subsystem or component needs, fill “niches”

BAD COTS

- Recent History
  - COTS products were touted as low cost “plug ins” to any architecture
  - “COTS glued together” even though the rage, was not effective or cost efficient

GOOD COTS

- Contemporary Answer – ISI’s answer
  - More robust COTS products, used appropriately and combined with good systems engineering practices, are highly cost effective and can be integrated into a flexible, extendable, maintainable architecture.
  - Requirements development can still be process driven, but the process can’t be rigid
  - Systems engineering plays a key role

INTEGRAL SYSTEMS

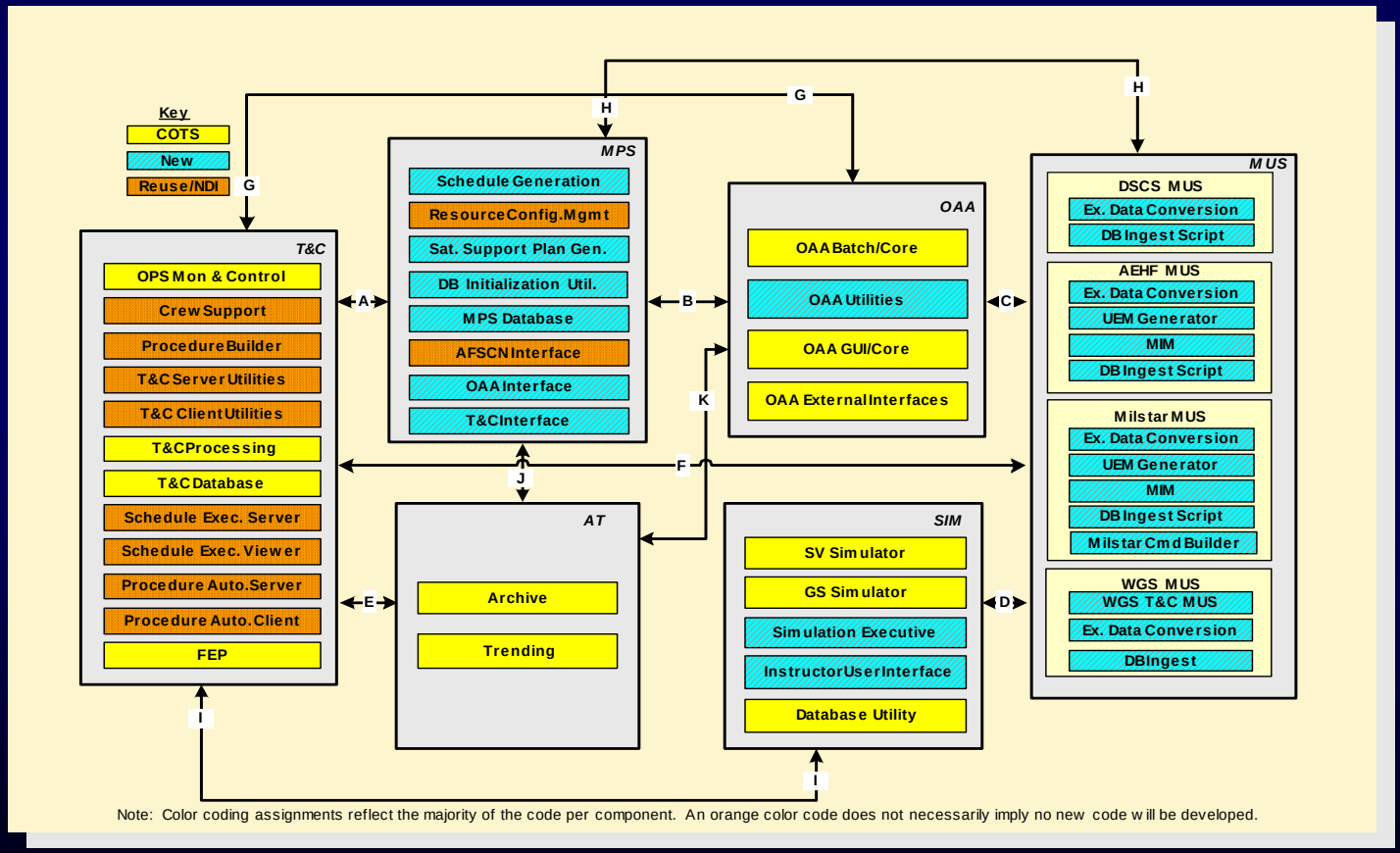
Validating COTS-based Architectures

- COTS allows rapid prototyping
  - Demonstration of capability early on of a small scale similar architecture is very effective
  - Factory developed performance tests of similar component quantities and loading is effective
- COTS allows fairly easy reconfiguring – “what if” configurations
  - Findings of performance tests can lead to validation, or drive selection of alternate architectures
- Future validation – formal subsystem and system test – is simplified
  - Its not necessary to test COTS components at code or unit level

INTEGRAL SYSTEMS

Creating and Validating COTS-Based Architectures

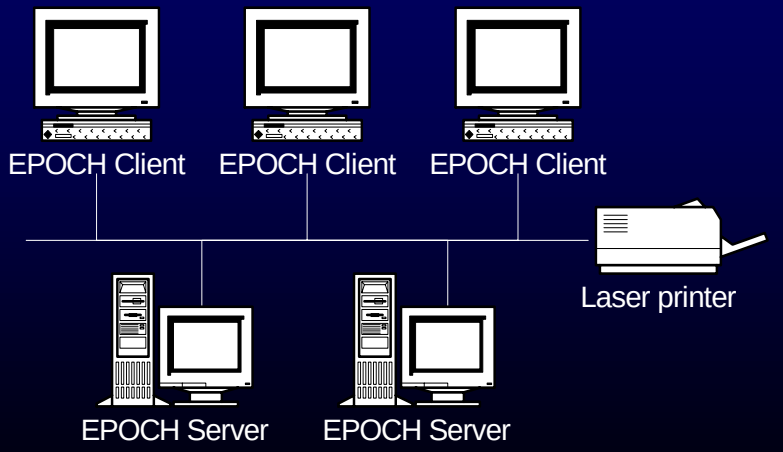
- CCS-C Architecture Example



INTEGRAL SYSTEMS

Typical ISI Architecture

- EPOCH Architecture
  - One of more servers and one or more user client workstations interconnected via LAN/WAN
  - Each server can control one or more satellites
  - Each client can connect to any/all servers
  - Spacecraft defined via database
- Open interfaces using industry standards allows integration with other COTS products



INTEGRAL SYSTEMS

CCS-C Example

- CSC-C used a multi-tired approach to testing
  - Initial performance
    - Initial performance data was available during contract demo phase
    - Performance data was available at CDR
  - Informal Subsystem and system testing
    - Components with custom code required more rigorous testing
    - Component level test done at contractor factory
  - Formal
    - Subsystem, inter-subsystem interface, and some system level testing done at factory
    - Preliminary end-to-end ops and performance testing conducted at factory, and that shaped the formal test on site
    - Subsystem install and regression testing done on site
    - System and external interface testing done on site

INTEGRAL SYSTEMS