

*Session 2 -- COTS and Legacy  
Software Integration Issues*



GSAW

Dennis Smith, Chair

Rhoda Novak, Co-Chair

February 26, 1998

# *Basic Structure of Session*

- Objective: identify major issues, current work and emerging solutions in COTS and legacy software integration issues
- Approach:
  - speakers develop basic themes
  - brainstorm primary issues
  - discuss one issue (cost) in more detail

# *Speakers Develop Basic Themes-*

## *1*



- Jay Costenbader, TSI TelSys Inc
  - “An Open Reconfigurable Computing Approach to Space Data Communications Processing”
- Michael Grier, Raytheon
  - “Methods and Technologies for a Reusable SGS Architecture”
- Arthur McClinton, Mitretek
  - “NOAA POES Engineering Network Prototype Status”

# *Speakers Develop Basic Themes-*

## 2

- Larry McQueary, Storm Control Systems
  - “The Selection of Middleware for Satellite Command and Control Systems”
- Bill Stratton, Integral Systems
  - “The NEAR Test and Mission Operations Ground System: Case Study in Rapid Development Using the EPOCH 2000 Spacecraft Command and Control Product”
- Paul Thoreson, L-3 Communications
  - “The Role of Scalable, Client/Server Technologies in Emerging Space Ground Systems”

# *Identification of Basic COTS and Legacy Integration Issues- 1*

- Cost issues (Chris Abts's USC model as a starting point)
- Ability to modify requirements to match COTS capabilities
  - A demo in the hand is worth a pound of requirements
  - Incremental prototypes erode requirements
  - Dynamic tension between acquisition requirements to use COTS and the best solution to be provided
    - importance of understanding the limitations of current generation of COTS before mandating COTS
- COTS-unique HCI versus common operator interface standards
  - Reducing systems cost versus operator training costs
  - Operators are more computer literate
  - Balance HCI standard versus standard operator interface

# *Identification of Basic COTS and Legacy Integration Issues- 2*

- Structure of the architecture and constraints
- Techniques for mining legacy assets
- Re-coding and re-hosting
- Long term maintenance
- Translators and emulators
- Middleware standards and products
- Product line issues

# *Identification of Basic COTS and Legacy Integration Issues- 3*

- Maintenance of systems where you don't own source
  - How does a single user get view into requirements and integrations created by others
- Lack of customer control over deliveries
- Language differences
- Vendor support for 10-20 year program life
  - Lack of sustained support for features in baselined COTS version
  - Tradeoffs for upgrading COTS baseline
  - Vendor viability

# *Identification of Basic COTS and Legacy Integration Issues- 4*

- Approaches to minimize breakage when substituting COTS products
- Network issues
- Planning and management issues
  - Risk
    - Using COTS to reduce risk
    - Unintentional risk added by COTS shortfalls
- Dormant Code (COTS unspecified code)
  - included in package without requirements
  - Undesired side effects

# *Identification of Basic COTS and Legacy Integration Issues- 5*

- The inability to predict systems behavior
  - Interactions among multiple COTS
    - e.g., with one stimuli, several COTS may be activated
- Resource contention issues with scalability
- COTS advertising versus prototype capability
- Verification and validation for COTS
  - Black box versus white box testing
- Version interdependency between COTS
- Tracking COTS state of the art
- Liability for COTS-related mission failure

# *COTS COST Discussion- 1*

- Every factor we discussed has cost impact
  - Consider using COTS within their limitations
  - Influence COTS vendors to handle
    - Verification and validation
    - Black box
    - Maintenance costs
- Government influence (and lack of) with vendors

# *COTS COST Discussion- 2*

- Need new cost models to handle COTS costs
  - Cost models should address cost factors that have variances and whether they are high, medium or low (Chris Abts's USC model)
    - Volatility is a big cost driver
    - COTS vendors may need to be periodically replaced and should be included in the cost modeling
    - COTS license costs for multiple hosts impact life cycle costs
    - Value (Return on Investment) -- cost versus benefit
    - Give up control of development and maintenance (COTS) versus full control over custom code
    - Maintenance is a major proportion of total life cycle cost
      - 10 to 20 year life of systems versus 18 month software technology development cycle

# *COTS COST Discussion- 3*

- Use COTS where it makes sense
  - Sufficient demand
  - Evaluate appropriate uses of COTS
- National systems have very high data rates
- Provide framework to allow COTS, custom and heritage components as required
  - how should acquisition authorities determine when to accept COTS with all its required and unused functionality
- Be sure you can live within the COTS capabilities (and lack of capabilities)

# *COTS COST Discussion- 4*

- Product line potentials
  - If there is only ragged legacy code and no COTS for the domain, it may be better to build custom code for new product lines
  - Modern systems are rarely developed totally from scratch
- Reuse when practical
- Use demos to preview actual COTS capabilities

## *COTS COST Discussion- 5*

- Is COTS volatility improving?
  - Market share impact of new features
  - Need easier to implement standards
- During system life it is sometimes not possible to replace one of the nodes of our systems (e.g. a satellite)
  - the node may outlive COTS
  - should we build cheaper, shorter-lived satellites