The Space Network Ground Segment Sustainment (SGSS) Project:

Developing a COTS-Intensive Ground System

Frank Herman, Linda Esker, Madeline Diep, Kathleen Dangle: Fraunhofer USA
Jeremy Jacobsohn, GMV
Rick Saylor, NASA GSFC
Constance Hoffman, General Dynamics
Agenda

• Project Overview

• Project Approach

• Studies and Results

• Experiences of a COTS Supplier

• Boundaries of Product and Project

• On-Going Issues

• Key Success Factors
Project Overview

**Project:** Space Network
Ground Segment
Sustainment (SGSS)

**Purpose:** Implement a new modern ground segment that will enable the NASA Space Network (SN) to deliver high quality services to the SN community for the future

**Key SGSS Goals:**
- Re-engineer the SN ground segment
- Enable cost efficiencies in the operability and maintainability of the broader SN.
Project Approach

• Consider a more COTS-based solution alternative to the current custom ground system.

• Prepared by conducting key studies:
  1. A replacement options study
  2. A COTS applicability study
  3. Quick studies conducted by several companies on how the Space Network ground system could be replaced with a new architecture within the existing facilities
1. Study Results-Replacement Options Study

• **Purpose:** Evaluate options for how the current functions could best be replaced, developing candidate operations concepts and architecture approaches and provided input to a COTS applicability study.

• **Results:**
  
  – Validated the feasibility of migration from serial to Internet Protocol-based network and analog to digital data processing and distribution
  
  – Provided a logical top-level grouping of functions/capabilities that would enable management of acquisition of SGSS system
2. Study Results-COTS Applicability Study

• **Purpose:** To conduct a survey on COTS availability/applicability for the modernized ground system.

• **Results:**
  – Provided recommendations for where COTS hardware and COTS software tools may be appropriate and which specific tools might apply
  
  – Identified risks associated with using those COTS products, and provided recommendations for how to mitigate those risks
3. Study Results-Architecture Studies

- **Purpose:** To task several companies to conduct quick study on how the SN ground system could be replaced with a new architecture within the existing facilities.

- **Results:** Concluded that the objectives of SGSS were achievable with technology generally available within industry with some custom extensions
  - Most provided scalable, open architecture with an emphasis on COTS, though some proprietary solutions were proposed
  - Proposed a number of COTS products that could be incorporated into the overall solution
  - Showed architecture approach can help with COTS integration
    - Using standards for interfaces and virtual machines (VM) help isolate product dependencies
Going Forward with COTS

- At RFP submittal, the SGSS SOW specifically required the contractors to use COTS hardware and software unless shown not in the best interest of the Government.
- SGSS Project is currently implementing a modern ground system
  - Little custom hardware
  - Many COTS/MOTS hardware and software suppliers.
  - Service Oriented Architecture (SOA) principles to enable software modules to be more easily updated or replaced in future planned technology refresh cycles.
  - System PDR completed
  - Element/Subsystem CDRs currently underway
- Gaining valuable experiences adapting and integrating many COTS products
  - Modification is required for unique COTS products such as TTC and Scheduling.
Experiences of a COTS Supplier: GMV

- One of the key COTS suppliers for SGSS
- Providing the SGSS project with COTS products for
  - Fleet and Ground Management (FGM) including the hifly® TTC system and the archiva data storage and trending system
  - Schedule Management (SM) including the flexplan planning and scheduling engine
- Participating in a full project lifecycle for development of product extensions based on the COTS products
Matching COTS to SGSS is challenging

- Original TDRS are highly specialized platforms
  - Many unique features compared to typical geosynchronous earth orbiting (GEO) satellites
  - No future market – can’t develop platform on investment dollars

- SGSS operations concept is highly integrated
  - Sophisticated concepts for software management not seen in most control center installations
    - Automated installation and upgrade of TTC clients and servers
    - System state managed by other systems (schedulable resources)
  - Operations include both satellite housekeeping and payload control
    - Simultaneous activities must be interleaved
• Matching COTS to SGSS is challenging (cont.)

  – Tension between open APIs and security requirements
    • Product has Corba and SOAP interfaces designed for open access
    • SGSS has complex security requirements to lock down access points
At the Boundary: GMV Product and Project

• The easy part
  – COTS TTC software has existing adaptation points
    • Scripting language to automate command and telemetry operations
    • Facility for computing telemetry-derived parameters
    • External API to exchange data with external systems
  – Some functions are entirely Project code
    • Conversion of legacy satellite databases and historical data archives
      – Perhaps would not have been needed if standards had been used in legacy systems
    • Conversion of existing paper or automated operational procedures

• The hard part
  – Mission unique requirements that need to fit ‘inside’ the product
    • Example: specialized command upload verification that requires tight binding between command formatting and telemetry processing
  – Requires code changes to COTS to enable custom code integration
At the Boundary: GMV Product and Project

- **Divide and Conquer**
  - Some features require the COTS to provide a new ‘hook’
    - Addition to existing external API (e.g. SOAP)
    - Library refactoring to provide an externally callable function or extension by sub-classing
  - Work is divided into:

<table>
<thead>
<tr>
<th><strong>COTS Product upgrade</strong></th>
<th><strong>Project specific extension</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Product</strong> change becomes a permanent part of the codebase</td>
<td><strong>Project</strong> code does not become part of the <strong>product</strong> codebase</td>
</tr>
<tr>
<td>- Required to maintain compatibility when <strong>product</strong> upgrades occur</td>
<td>- Is deliverable to the <strong>project</strong> including all required libraries and build infrastructure</td>
</tr>
<tr>
<td><strong>Product</strong> change is funded by internal investment</td>
<td><strong>Project</strong> code is funded by end customer</td>
</tr>
<tr>
<td>- Vendor maintains Intellectual Property (IP) rights to <strong>product</strong> codebase</td>
<td>- Customer maintains IP rights and can maintain/extend independently</td>
</tr>
</tbody>
</table>
On-Going Issues

1. Each COTS in the system has a set of prerequisites for development and runtime. These may conflict with the prerequisites of other COTS, or with licensing policies.

– System-level runtime and build environments need to be nailed down early in the project, and maintained as development proceeds. This includes:
  i. Base OS (to the specific version and processor architecture choice)
  ii. Optional packages from the OS distribution
  iii. Third-party packages in the build environment
  iv. Third-party packages in the test environment
  v. Third-party packages in the runtime environment

– Compatibility with anticipated build, runtime and licensing environments should be a factor in COTS selection
  i. This may need to be iterative, because the set of candidate COTS may influence the selection of environment
2. Nomenclature

- COTS tools have a set of terms used in UIs and documentation
- These terms may differ from terms in use in legacy systems or other COTS
- Modification of the COTS would cause incompatibility with future releases, or inconsistency with documentation
- Where small number of users are involved can be handled with system documentation and training

Example: What WSC calls a schedule request is represented in flexplan by an event

- This is visible only to a small number of planner/schedulers at WSC
- Space Network Users can continue to use their existing names

"What's in a name? That which we call a rose By any other name would smell as sweet."
Key Success Factors

• Teamwork
  – SGSS project team needs wide expertise
    • Subject Matter Experts on TDRS and legacy ground system, experts on individual COTS, components, custom developers,…
    • Systems Engineering to tie it all together
  – COTS suppliers have critical roles
    • Provide feedback during requirements allocation
      – Systems engineers need to understand how to use COTS optimally or where a requirement should not be allocated to COTS
    • Advise design team on use of COTS extension points
    • Offer services for project-specific development and testing
      – Cost effective for project due to training and experience with COTS product

• Early prototyping or hand-on interaction with the COTS
  – Can’t wait until after the design is complete to procure COTS
  – COTS spec sheets can’t always provide all the needed information
Questions?
Contact Information

• Frank Herman:
  – fherman@fc-md.umd.edu

• Linda Esker:
  – lesker@fc-md.umd.edu

• Jeremy Jacobsohn:
  – jjacobsohn@usa.gmv.com

• Rick Saylor:
  – Richard.S.Saylor@nasa.gov

• Constance Hoffman:
  – Constance.Hoffman@gdc4s.com
## Acronyms

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>API</td>
<td>Application Program Interface</td>
</tr>
<tr>
<td>CDR</td>
<td>Critical Design Review</td>
</tr>
<tr>
<td>Corba</td>
<td>Common Object Request Broker Architecture</td>
</tr>
<tr>
<td>COTS</td>
<td>Commercial Off-the-Shelf</td>
</tr>
<tr>
<td>FGM</td>
<td>Fleet and Ground Management</td>
</tr>
<tr>
<td>GEO</td>
<td>Geosynchronous Earth Orbiting</td>
</tr>
<tr>
<td>IP</td>
<td>Intellectual Property</td>
</tr>
<tr>
<td>MOTS</td>
<td>Modified Off-the-Shelf</td>
</tr>
<tr>
<td>OS</td>
<td>Operating System</td>
</tr>
<tr>
<td>PDR</td>
<td>Preliminary Design Review</td>
</tr>
<tr>
<td>SGSS</td>
<td>Space Network Ground Segment Sustainment</td>
</tr>
<tr>
<td>SM</td>
<td>Schedule Management</td>
</tr>
<tr>
<td>SN</td>
<td>Space Network</td>
</tr>
<tr>
<td>SOA</td>
<td>Service Oriented Architecture</td>
</tr>
<tr>
<td>SOAP</td>
<td>Simple Object Access Protocol</td>
</tr>
<tr>
<td>TTC</td>
<td>Telemetry, Tracking, and Command</td>
</tr>
<tr>
<td>TDRS</td>
<td>Tracking Data Relay Satellite</td>
</tr>
<tr>
<td>UI</td>
<td>User Interface</td>
</tr>
<tr>
<td>VM</td>
<td>Virtual Machine</td>
</tr>
<tr>
<td>WSC</td>
<td>White Sands Complex</td>
</tr>
</tbody>
</table>