GSAM 2011 Implementing Shared Capabilities

Ground Software Integration in Various Bus Architectures: Lessons Learned from Recent Experience

© GMV, 2011
Who Are We?

GMV is....

- Privately-owned, family business (woman-owned)
- Established in 1984 with affiliates in the US, Europe and Asia
- ISO 9001 certified, CMMI level 5 certified for SW development
- Space News Top 50: #48 for 2 years in a row
- #1 Commercial telecom ground system supplier in the world
- Only company to sell ground SW to space institutions around the world (NASA, NOAA, USGS, CNES, ESA, Eumetsat, Roscosmos, ISRO, ETRI)
- GMV ground systems deployed to 26 countries on 6 continents
Recent Experience

- 5 GEO fleet migrations successfully completed
- Largest independent GEO fleet migration in the world ever done was successfully completed by GMV in 2008
- NASA Goddard’s GMSEC
  - Interoperability standards with wide application
- Satellite manufacturer R&D
  - Test benches to demonstrate spacecraft compatibility
- NASA SGSS and GOES-R programs
  - GMV COTS products fit in to larger architectures
  - SOA is required
  - Dovetails with product development of upcoming GMV COTS versions
Systems provide “services” to each other via bus I/F based upon XML
Replace HW/SW easily

Events, warnings, logs, status for all G/S elements provided to SA
Distributed Architectures: Corba

Widely used in:
- Banking & Finance Online account access
- E-commerce
- Network management
- Hospital Patient Record Management
- Entertainment pay-per-view
- Spacecraft control centers!

Limitations:
- Significant learning curve
- Complex object life cycle
- Requires bridging to older legacy systems
- Network difficulties (firewalls, non transparent addresses)
- Mobile environment limitations (changing and unreliable network addresses)
Distribution Architectures: Web Services

- B3C Activity to layer additional features above SOAP and XML
- Service discovery
- Security
- Web Services Description Language (WSDL)
- Can use other transports
- RPC (like Corba)
- Message Bus
- Verages the structure of the web
- HTTP(S) communications
- URI resource location
- REST techniques aid caching
Graduated Spaces: Learning Curve

- Message Bus
  - Relatively simple
- Web Services
  - Requires many technologies
  - Skills are widely available
- Corba
  - No way to start small
  - Skills availability may have peaked
Upgrade Spaces: Binding

- **Message Bus**
  - Runtime
  - Infrastructure doesn’t enforce

- **Web Services**
  - Runtime
  - WSDL allows run-time check of data structures

- **Corba**
  - Build time
  - Early detection of type errors
  - Often circumvented with ‘any’ type
Scale Spaces: Scalability

- **Corba**
  - RPC based point to point

- **Message Bus**
  - Scales well within the LAN

- **Web Services**
  - Scales to the world
  - Requires attention to caching and load balancing
  - REST: ‘get’ requests cacheable
Tool Maturity

- Web Services
  - Changing rapidly
  - Growing provider base

- Corba & Message Bus
  - Flight proven
  - Provider base is shaken out and stable
Technical Enablers

Are the legacy systems built from distributed components?

Are the interfaces fixed and known?

-plumbing is feasible
Technical Disablers

Some algorithms are difficult to replicate in new technologies

- Accounting sensitive feedback loops
- Carefully tuned rule bases
- Black Box’ components with lost source code
- The end of the interface is immutable
- The ground system and flight software were developed together
- Now the flight side is out of reach (literally!)

We’re no smarter than “Those Who Have Gone Before”
Integration Obstacles

Many **barriers** prevent quick deployment of new technologies into existing architectures.

**Long missions** (e.g. typical GEO is 15 years). Difficult cost/benefit ratio for technology upgrade.

Operators are **reluctant to lose features** in transition to a new system. *I want everything I have plus a lot more...*

Large **variability** of requirements across missions.

- What works well for one may fail for the next
- Scalability issues (e.g. single satellite vs constellations)
- Difficult to create a ‘generic’ technology that will suit all

Difficult **deployment** of systems and quickly evolving OS.
“green fields”
Existing operations centers have frameworks in place
Every component has a legacy tail – otherwise it wouldn’t be selected
Team skills are long-lead items
Choose bridge points
Prefer
  - Minimize # bridges
  - Stable component boundaries
Avoid
  - Latency-sensitive interfaces
  - Introducing critical failures
Inclusions

“one right answer”
Message Bus for steady-state flows (e.g. telemetry frames)
Web Services for highly variable and scalable client loads
Corba for tightly bound internal transactions
Legacy is important
Existing equipment and systems may have infrastructure in place
Baseline COTS may have a preferred infrastructure
Smart vendors are ready to support others as well
Maintenance staff may be highly skilled with an infrastructure
dge as required
Systems of systems may require combinations of architectures
Thank you