GSAW 2006 - LA SPIN

Evolution or Revolution

Dealing with Ground Segment CSOS (Complex Systems of Systems) in the Real World

N. Peccia, ESA
OUTLINE

➢ The Ideal World to develop a new piece of Software

➢ What is going on:
  ➢ International Level - The CCSDS Monitoring & Control Services
  ➢ European Level - The European Technology Harmonisation on Ground Software Systems
  ➢ Organization Level - ESA Initiative: EGOS

➢ The Real World - How to deal with CSOS

➢ Conclusions
How did we do in the past?

 legacy System

 Ca. 15 years

 New System

 Ca. 15 years

 OOA / OOD

 C++
The Ideal World to develop a new piece of Software

- If I need to implement a new Data System for a Ground Segment from scratch
- I will select the following (towards a platform independent model, no technology considerations):
  - MDA - Model Driven Architecture
  - SOA – Service Oriented Architecture
  - A consolidated SDE (Software Development Environment)
    - Requirements Management Framework
    - Configuration Management Framework
    - Modelling Framework
    - Validation Manager
Model Driven Architecture

Requirements capture
- High level model
  - Enterprise architect
  - Magicdraw

Requirements refinement
- Detailed model
  - Enterprise architect
  - Magicdraw

Model validation
- Executable model
  - Kennedy Carter
  - Rhapsody
  - Telelogic TAU

Validation
- Validation tests
  - SilkTest (GUI)
  - IDATG
  - Telelogic TAU

Integration testing
- Integration tests
  - Parasoft tester
  - Telelogic TAU

Testing framework
- Parasoft tester
  - JUnit

Unit testing
- Unit tests
  - Parasoft tester
  - Telelogic TAU

Unit tests
- ParaSoft tester
  - Telelogic TAU

Code generation
- Skeletons
  - Enterprise architect
  - Code generation
  - Kennedy Carter
  - Rhapsody
  - Telelogic TAU

Testing framework
- Transition between development phases
- (semi) automatic conversions
- Assistance from special tools
Model Based Development

- The development is centered around the model
  - Ensures that model and code are always aligned
  - Provides a “technology-independent” basis: never will be obsolete

- The model is validated in early phases
  - Ensures that the final product is in-line with requirements
  - Avoid costly errors in late development phases

- Large amounts of the code are generated automatically
  - Reduces human errors
  - Enforces reuse
  - Coding language can be selected based on specific needs. Several languages can coexist (e.g. Java on GUIs for automatic testing, C++ on core).

- Testing is supported by special tools
  - GUI testing: user emulation allows accurate repeatability of the tests
  - Unit and integration testing: test corpus is generated from the code skeleton
  - Validation testing: test cases derived from the model
Model Based Development

- Helps the designer to achieve clarity of vision.
- Allows for communication of ideas from a common context.
- But
  - How close is the implemented system to the model?
  - How do we achieve a tight coupling of implementation and model?
- And
  - Central representation with various views.
    - Visibility of changes?
  - Decentralized models
    - Cross model knowledge.
    - Large number of links
Software Development Environment

- Requirements Manager (DOORS)
  - SRS SW Req Spec

- Configuration Manager (Synergy)

- Data Model (Enterprise architect)
  - XMI models
  - Image (Word)
  - Trac. Matri (Word)

- VM (Validation manager)
  - Word reports

- Word reports

- Siemens ADTG ?
- ART Test Commander ?

- Version and change control

- Automatic Test schedules

- Automatic Test schedules
SDE Framework in UML
Modelling Framework

- Requirements (Enterprise Viewpoint)
- Information / Data (Information Viewpoint)
- I/Fs / Services and Components (Functional Viewpoint)
- Protocols, Bindings, etc. (Communications Viewpoint)
- Deployment (Physical Viewpoint)

- Based on RM-ODP
- Applies MDA and use UML-2

Standardised Data Models
Standardised Services
Technology Selection

- **Target platforms:**
  - Common Base Code
    - SUN / Solaris
    - PC / SuSE Linux (SLES)
  - Languages:
    - C++
    - JAVA
  - Adaptive Communication Environment (ACE) or CIAO (Component-Integrated ACE ORB)
    - CCM CORBA Component Model
    - Messaging
    - Inter Process Communication (IPC)
    - Etc.
  - Eclipse (MMI framework)
International Level

The CCSDS M&C Services
Service Oriented Architecture
Example Distribution of Mission Operations Functions
Example Distribution of Mission Operations Functions

MO Domain Specific Applications

Plug-in Applications

SM&C MO Service Framework

Technology Adaptors

Technology Specific Infrastructure

Consumer Application

Provider Application

SM&C: Mission Operations Services

SM&C: Common Services

SM&C: Protocol
Message Exchange; File Transfer; Mail

Infrastructure Services
Message Exchange; File Transfer; Mail

Communications Services

Interoperable Protocol (GPIO)

SAP: Service Access Point
Service Tunnelling

SLE Services:
- SLE Service Management
- SLS Transfer Services [Forward/Return]
- Open [Super] Service
...

Space Link Services

SM&C Common Protocol Service

SLE / Mission Operations Services:
- Radiometric [Tracking & Ranging]
- GS [Space Link] Monitoring
...

Mission Operations Services:
- SM&C Core Service
- Time & Location
- Scheduling & Automation
- On-board Software Management
...
Overview: Mission Operations Services
Potential Ground Segment Services
The Service Model

Service Provider

Information Model

Service Clients:
Ground Segment Applications

Mission Control System

Service Interfaces

Archive

CIM Services

Core Services

Historical Services

Management Services
Mission Operations Service Framework Layer

Mission Operations Services
- Core, Time, Software Management …

Common Services
- Directory, Replay Control
- Generic Interaction Patterns

SM&C Protocol
- Messaging
- File Transfer
- Mail

Consumer/Provider

Mission Operations Service Framework

[Images and diagrams showing the framework layers and services]
Layered Services & Operations Languages

- Parameters
  - Commands
  - Alerts
- Orders
  - Events
  - Tasks
  - Activities
- Orbit Vectors
- Ranges
- Manoeuvres

Service Provider [Application]

Service Layer

Distribution Layer

Network Layer

Operating Language Layer

Service Consumer [Application]

Information Model

M&C

Scheduling

Orbital

Expressions

Conditions

Scripts

Display

Automation

Planning

Analysis
European Level

The European Technology Harmonisation

on

Ground Software Systems
We focus on Cross-sectorial strategy for Ground Segment
Strategy Landscape

- Cross Support Interoperability at Ground Segment Components (high level of granularity)
- European Harmonisation Interoperability at Ground Segment Subsystem (medium level of granularity)
- Software Re-usability at Software Middleware Components (low level of granularity)

International
Via CCSDS / ECSS / OMG standards

European
Via ECSS standards
Plug & Play Subsystem Components

Organization
Plug & Play Middleware Components
Our Roadmap

<table>
<thead>
<tr>
<th>T</th>
<th>T+1 year</th>
<th>T+2 y</th>
<th>T+3 y</th>
<th>T+4 y</th>
</tr>
</thead>
</table>

System & Requirements Engineering

Interface Definition and Validation

Ground System SW Products Selection / Re-engineering

Ground Segment SW System description
Reference Architecture, Requirements, Services, External ICD List

Set of European compliant Products

ECSS: European Cooperation for Space Standardisation
Ground System SW: The ECSS-70 View

Satellite Simulator
- Ranging
- Tracking
- Time Cal
- M&C
- Data storage
- Data Distr.
- Uplink Data Processing
- Downlink Data Processing

Ground Station

Flight Dynamics System
Mission Planning & Schedule
Commanding
OBSW Management
Control Procedure Generation
Control Procedure Execution
Product Generation
Monitoring
Performance Evaluation
Data Archive
Operational Database Management
OBSW Maintenance
HCI

Electrical Ground Support Equipment: EGSE

GSAW 2006 – LA SPIN
N. Peccia ESA
29th Mar 2006
A typical Satellite EGSE

- SIMULATOR
- Front End Equipment
- Special Checkout Equipment
- TEST SUPERVISOR
- Spacecraft
- External Systems
- On-Board SDE
- Mission Control System
- Mission DB

GSAW 2006 – LA SPIN
N. Peccia ESA
29th Mar 2006
The Principle of Harmonization

French Space Agency
- CNES

German Space Agency
- DLR
- ESA

COTS Vendor 1
- EADS

COTS Vendor 2
Interface Requirements (MPS-M&C-FDS)

The three Ground Control interface requirements are developed in coordination.

**MPS-M&C**
Allows MPS to send operational schedules for execution on the M&C or elsewhere. M&C provides feedback to MPS on the success or failure of scheduled activities.

**MPS**

**FDS**

**FDS-M&C**
- Allows flight dynamics to monitor and exert control over the spacecraft flight.
- Additionally, it allows M&C to use and distribute FDS data.

**FDS-MPS**
- FDS receives navigation change requests from MPS and then calculates the maneuvering schedule needed to perform them. This schedule is then supplied to MPS.
Organization Level

The ESA New Initiative

EGOS

ESA Ground Operations Software
The vicious cycle of R&D

Studies
- Exploration of new Technologies and Functionalities
- Concept validation

Infrastructure
- Generic systems re-used for different missions and applications
- Long term mission profile

Standardisation
- Definition, production and reviewing of European and International standards (ECSS, CCSDS)

Customisation
- Mission Specific configuration, adaptations, extensions

PROTOTYPES

STANDARDS

PRODUCTS

STANDARDS
The EGOS Initiative

- To allow ESA to be an active and leading player in the European harmonisation process
- To examine critically our existing Ground Segment software infrastructure and to prepare it to fit in the harmonisation framework
  - Functions -> OK
  - Interfaces -> need to be adapted
- But need for harmonisation even within ESA from a technology viewpoint
  - Technical solutions / products (COTS, Open Source)
  - Development methodology
  - Maximise common software basic components (e.g. middleware)
What is EGOS?

- **ESA Ground Operation Software**
- **-» European Ground Operation Software ??**

- It is at the same time a paradigm, a concept and a project
  - Future infrastructure for ESA ground segment systems
  - Aims to standardise and harmonise existing systems
  - Improve interoperability issues
  - Evolutionary approach required due to size of existing code base
Rational: If it is not broken why fix it?

- Heterogeneous operating systems, i.e. lack of common approach
- Heterogeneous hardware platforms
- Different HCI (Human Computer Interface) look and feel
- No standard network and communication services
- No standard language or protocol for data interchange
- Lack of common standard for event and log messaging
- Lack of common data access across subsystems
  - Files and databases
  - Metadata models used
  - Different internal representations

- This is a problem in all existing Ground Segment software infrastructures!!
Architectural framework

- Common layered system approach
- Common set of services
- Commonly usable new / adaptive middleware software tools
- Reduction of costs for new development (by using a common service model)
- Interoperability across systems
- Federation of products
- Support of autonomous operations
- Common software maintenance policy
- Reduction of the familiarization efforts of both the users and developers teams
- Support to European Industry Harmonisation by giving them free licenses for EGOS products to give a stronger competition position worldwide
Service Management Framework

- Provides encapsulation layer (keep what we have as infrastructure)
- Exposes services to external users/systems in standardised manner
- Controls access to exposed services
- Interfaces to internal services via drivers that handle required protocol conversion
A target Architecture

MAS: Mission Automation System
EDDS: EGOS Data Disposition System
NIS: Network Interface System
EMS: ESA Ground Station Management System
STC: Ground Station Station Computer

MAS | EMS | EDDS | STC | NIS | Others
---|---|---|---|---|---
Ancillary Systems

SMF
Service Management Framework

Driver

GUIs

TM/TC Processing Components

Driver

Common Services

Driver

Common Libraries

Events
Logging

Service
Directory

Configuration
Management

Others

EGOS Framework (Basic Services)

MAS: Mission Automation System
EDDS: EGOS Data Disposition System
NIS: Network Interface System
EMS: ESA Ground Station Management System
STC: Ground Station Station Computer
The Real World

The current Ground Segment CSOS

Evolution or Revolution
CSOS: Our CONTEXT

- Satellite Simulator
  - Ranging
  - Tracking
  - Time Cal
  - M&C
  - Data Storage
  - Data Distr.
  - Uplink Data Processing
  - Downlink Data Processing

- Ground Station
  - Flight Dynamics System
  - Commanding
  - Monitoring
  - Operational Database Management
  - Data Archive
  - OBSW Maintenance
  - Control Commanding
  - Control Procedure Execution
  - Performance Evaluation
  - Product Generation

Electrical Ground Support Equipment: EGSE
## Size of our Systems

<table>
<thead>
<tr>
<th>Application</th>
<th>Release</th>
<th>LoC</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCOS-2000</td>
<td>4.1</td>
<td>527,968</td>
</tr>
<tr>
<td>TDRS</td>
<td>2.1</td>
<td>21,286</td>
</tr>
<tr>
<td>MCCM</td>
<td>3.0.7</td>
<td>19,769</td>
</tr>
<tr>
<td>WebRM</td>
<td>3.2</td>
<td>41,544</td>
</tr>
<tr>
<td>GDDS</td>
<td>1.5</td>
<td>30,000</td>
</tr>
<tr>
<td>NCTRS</td>
<td>10.0.1</td>
<td>188,924</td>
</tr>
<tr>
<td>SLE API</td>
<td>2.0.3</td>
<td>101,819</td>
</tr>
<tr>
<td>ESG</td>
<td>2.1</td>
<td>31,226</td>
</tr>
<tr>
<td>TMTCS</td>
<td>20.12.2004</td>
<td>143,331</td>
</tr>
<tr>
<td>GSSC/GMMI</td>
<td>2.4</td>
<td>254,498</td>
</tr>
<tr>
<td>MCM4</td>
<td>1.3</td>
<td>33,113</td>
</tr>
<tr>
<td>STC2</td>
<td></td>
<td>630,000</td>
</tr>
<tr>
<td>CRS</td>
<td>1.1.3</td>
<td>12,000</td>
</tr>
<tr>
<td>Simsat</td>
<td>3.0 FA</td>
<td>135,000</td>
</tr>
<tr>
<td>PSS</td>
<td>1.2.2 Issue 2 Java</td>
<td>36,000</td>
</tr>
<tr>
<td>Emulator Suite</td>
<td>1.4</td>
<td>292,600</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td></td>
<td><strong>2,499,078</strong></td>
</tr>
</tbody>
</table>

Cost of SW assets = 30 MEuro
Characteristics of our CSOS Legacy Systems

- Heterogeneous Operating Systems, i.e. lack of common approach
- Heterogeneous Hardware platforms
- Different HCI (Human Computer Interface) look and feel.
- No standard network and communication services
- No standard language or protocol for data interchange
- Lack of common metadata model
- Lack of common standard for event and log messaging
- Lack of common standard for data access (files and databases)
- Variety of databases used across subsystems
- Different internal representations for data
- Lack of common approach to System Monitoring and Control
- Lack of common approach to Security
- Lack of common approach to fault management
- Lack of common approach to Configuration and system initialization
- Lack of unique software maintainability approach
- Lack of isolation of software systems from OS / COTS to improve portability
Characteristics of our CSOS Legacy Systems

- Lack of clear isolation between components of a subsystem
- Lack of synergy across developments
- Proliferation of test tools to support the validation of the various subsystems.
- Our CSOS from the SDE point of view
  - Requirements are in Word Files, Test Plans in MS-Access (in-house developed tool) + automatic generation of test plans
    - Tool can not cope with increase in size and complexity of the SW
    - Problems with associated test data and configuration files
    - Mapping between requirements and test plans is not straightforward
    - Link between SPRs and test cases is not done automatically
    - Not unified test plan, but a large collection of unconnected test plans
  - Architectural design is supported by models
    - Difficult to maintain as SW evolves
    - Architectural conception at top level is often changed during the implementation phase
    - New modelling methods and languages (such as UML 2.0) makes some existing models obsolete and un-maintainable
    - Mapping between requirements and top level architecture components is sometimes obsolete
Other Considerations

- The rate of technology changes is very high
  - Will SW implemented today run on the same platform tomorrow?
  - Will a new language appear?
- Can my legacy system be opened via exposing services?
  - Not showing the processes (normally described via UML) but only the services
How to deal with all the changes?

Corrective Maintenance

Towards MDA/ SOA/ SDE

Towards European Technology Harmonisation

Towards CCSDS M&C Services

Towards EGOS

Perfective Maintenance

New Functions

LEGACY CSOS

TO BE DONE ON EVERY DATA SYSTEM

New CSOS
Other Considerations

- Which is the best approach: Evolutionary or Revolutionary?
  - It depends on the organization and its infrastructure cycle wrt technology available
  - An analysis of your CSOS stability is needed

- For compliance to standardised services and Interfaces
  - Does the organization keep ICDs / APIs? or Does the organization go for a whole scale re-engineering?
  - One solution is that organizations adapt their existing infrastructure where it exposes a given interoperable interface.
  - It is not necessary at the beginning to modify internal architectures, only to build adapters for exposed interfaces.
  - The advantage is that next time the adapter can be re-used

- In the longer term, when systems become obsolete the new generation is developed according to the standards and no more adapter is needed.
Don’t forget R&D

excessive CSOS infrastructure cost + lack of innovation

= loss of competitiveness

- make R&D a priority
- Increase coherence
  - among studies / R&D activities
  - between R&D and infrastructure activities
A Phased Approach

Isolation of middleware tools at system level

Selection of best candidate

Adaptation of best candidate to EGOS framework

Re-engineering of systems with best candidates

System A

System B

System C

SW Middleware Tool X

SW Middleware Tool Y

SW Middleware Tool Z

System A'

System B'

System C'

GSAW 2006 – LA SPIN

N. Peccia ESA
29th Mar 2006
How to evolve every Data System

Each cycle includes:
- Corrective / adaptive / perfective maintenance
- New functions
- Towards EGOS: replacement of Low level / Middle Level and / or MMI framework
- Adaptation of interfaces to new available standards

Changes wrt SDE / MDA / SOA
- Revision / upgrade of models via reverse eng. using source code as reference
- Upgrade mapping req. / architecture
- Upgrade mapping req. / test plans
- Import req. in Req. management Tool
- Deploy test plans in Validation Manager FW
Our Suggested Approach

- **Evolutionary**
  - Large existing code base \( \sim 2.5 \times 10^6 \) LoC

- **Development of new systems has to continue**
  - Try to minimise impact of future changes on these developments by using new concepts and available frameworks (low level, middle level, user desktop)

- **Approach depends on the appropriate selection of components**
Conclusions

- Our legacy CSOS consists of a large set of independently developed Systems
- We must continue to support and maintain this CSOS because our current missions are using them as their Baseline
  - And the users are conservative and reluctant to change this “well working” CSOS
- Infrastructure being developed nowadays will be deployed in 2 or 3 years.
- In order to minimise impact, ESA is following (and suggesting) an evolutionary approach
- Let’s see how far we are in 2 or 4 years