Smooth Transition and the ILT Experience

EGSE Systems in ILT/IST as a Whole

Steve Guest

Scope by Lesson

- LessonH9: One system to support all mission phases ("Smooth transition")
- LessonH30: The CUS provided a powerful instrument commanding system
- LessonH46: Smooth Transition--the "bad"
- LessonH55: MIB handling
- LessonH93: The EGSE and the HCSS
- LessonH94: EGSE Framework
- LessonH95: Selection and use of SCOS-2000.
- LessonH97: Transition: The Rough and the Smooth
- LessonH105: Managing a large system with smooth transition
- LessonH106: Strengths and weaknesses of CUS
- LessonH109: ILT/IST commanding environment
- LessonH133: Early start of Science Ground Segment development
- LessonH156: There should have been one common MIB ingestor
- LessonH157: Error injection in Simulators
- LessonH168: Telemetry parameter handling

The Smooth Transition concept

- View from “outside” ILT:
  - This was very successful and is recommended to future missions. (LessonH9)
- Views from ILT participants:
  - Generally a good concept but a few things to consider.
  - The use of the HCSS for ILT was a good concept overall but led to a lot of pain and additional effort for the ICCs early in development. (LessonH97)
  - The concept of smooth transition has indeed been of great benefit to the project, starting with the instrument team being capable to using the same uplink code form ILT to Routine. However one should not forget the enormous burden we've had to go through during the IST. (LessonH46)

The bumps in the transition

- Difficult to plan the development:
  - Early selection of tools was required e.g. Versant
  - Development schedule not synchronized with instrument schedules (might be impossible).
  - Very different instrument schedules and ILT starts…
  - All components should be ready for the first… (they weren't)…
  - Leading to divergent developments.
- Issues with system components:
  - Could be unstable and/or poorly documented.
  - Reluctance to redesign problematic early s/w or prototypes in the name of backward compatibility.
- Data products from ILT incompatible with pipeline products that were defined later.

Relationship EGSE-HCSS

- EGSE Working Group separate entity from HCSS Management Group.
  - Some developments under EGSE WG management (EGSE router, CDMS simulator). Test control half in one and half in other?
- Some things not a snug fit with “HCSS concept”:
  - Some essential EGSE components did not fit with Java development e.g. CDMS simulator (wholly), test control (partly).
  - Several wrappers had to be written in order to fulfil the uplink format needed by Industry (tcl-based).
- QLA systems under individual ICC management.
  - Some framework components under HCSS.

EGSE Systems

- Common EGSE hardware procurement (by PACS).
- Computer hardware is difficult to maintain over the period we have had to.
  - Ancient systems still running.
  - It was risky to maintain so much equipment that was no longer supported, although we have largely 'got away with it' .
- Both development plans and system designs need to take this life cycle into account, and allow for the need to upgrade or replace machines, operating systems and development environments.

Components of the EGSE (scope)

- SCOS-2000 + TOPE
- EGSE router
- Versant database
- MIB
- CUS
- Telemetry ingestion
- Test control
- binstruct + spire_param
- QLA systems
- HCSS framework components (access, logging, configuration)
- CDMS simulator
- TCH/OOL
- Miscellaneous tools (e.g packet display)

What would have come in handy

- Consistent and integrated configuration/versioning control e.g. TOPE scripts, CUS scripts, MIB.
- DP was not originally in smooth transition scope:
  - "The handicap in ILT was…primarily lack of HIPE… many non-HIPE off-line analysis tools were built in ILT, which were never taken over into standard processing."
- Automatic monitoring and restarting of processes.
- Better message logging and configuration/properties handling "out of the box".
- Integrated test log and note tool, avoiding fragmentation of information (could we have used a Wiki for this then?)
- Support for the simultaneous use of various models (development/flight/spare)

SCOS-2000 + TOPE

- Thorough selection process with ESA and all Herschel/Planck instrument teams.
- The necessary support and functionality was eventually provided but it was not always a smooth process.
- Environment during IST was good and efficient…
- TCl/TK interface to SCOS-2000 was a pain and it failed many times, especially in batch mode
- Would have been better to do the commanding through the CUS and have SCOS-2000 understanding the high level language or have a clear interface for it.

SCOS interfaces

- EGSE router & gateways
  - Nobody bothered to mention this, perhaps because it worked so well?
  - But why was the original SCOS router rejected and this one developed instead?
  - This module is fully reusable and works with current SCOS.
- Test control
  - Bridged “the gap” between SCOS tcl-based procedures and HCSS/Versant database.
  - Could be tricky to install and configure.

HCSS database

- Versant database
  - ILT/IST/Ops databases never integrated as intended
  - Some problems with locking in early days
- Telemetry ingestion
  - Worked pretty reliably
  - Reusability?

MIB & telemetry handling

- The idea of having this common database was good and helped in smooth transition, but it was not perfect.
- There should have been one common MIB ingestor, instead of the three we had for Herschel.
- The HSC software required two ingestion procedures for the MIB, one for uplink and a completely separate one for downlink.
- Industry used the MIB for spacecraft HK in a way that was not compatible with HSC use (notably extensive use of so-called synthetic parameters).
- Configuration control issues.
- Downlink services for decoding telemetry never used; instead binstruct developed; SPIRE had its own system.
- In retrospect maybe the SPIRE/binstruct systems should have been merged.

CUS

- (Also covered under uplink)
- Configuration control/versioning issues.
- The CUS as a language was good.
- Self-cooked language instead of existing language (e.g. python) - limited data types makes unreadable code (parameters packed in arrays)
- Facilities for a proper test harness should have been provided.
- In principle could be used on future missions.

QLA systems and tools

- QLA systems a major development
  - Three very different systems.
  - Mixed success in common framework. Different communication models used (e.g. dataflow only HIFI); DataSelector from “access” module used by all three.
  - Reusable QLA framework could probably be achievable.
- PacketDisplay
  - SPIRE tool, other ICCs may have had similar.
  - Reusable with a small amount of work.

CDMS simulator

- SPIRE contribution to common EGSE development.
- Spacecraft computer simulator based on 1553 bus.
- Real time hardware/software system using 1553 card.
- Low-level code in C++, user interface in LabView.
- Runs on Windows NT 4 (only)
- Outside HCSS configuration/versioning control.

TC History/Out of Limits

- This data was to be attached in the Versant database (part of linking uplink/downlink).
- These modules had a lot of discussion in HGSSE meetings.
- Significant development effort.
- Used sporadically rather than consistently in ILT.
- Not used at all in operations.
- Problem was performance?
- Did anyone miss not having this in the database?

OBS

- Where does the development and testing of the on-board software fit in? Is it covered anywhere else?

Lessons Digest