THE ISO DATA REDUCTION LESSON: PIPELINE AND INTERACTIVE ANALYSIS ARE COMPLIMENTARY SYSTEMS

S. Ott[1]

ISO Data Centre, Astrophysics Division, ESA, Villafranca del Castillo, P.O. Box 50727, 28080 Madrid, Spain

ABSTRACT

Nowadays, 30 to 60 % of the overall budget for ground based observatories is spent on software. For ISO it was less than 10 %.

We will argue that the existence of a pipeline *and* an interactive analysis system is essential for the success of an observatory mission and outline the respective merits and user communities of both systems.

We will review the history of ISO's pipeline development approach, outline the problems encountered and the lessons learned.

Finally, we will indicate a road map for cost-efficient pipeline development.

Key words: ISO, Software, Data Analysis

1. INTRODUCTION

ESA's Infrared Space Observatory (ISO) is an astronomical satellite, operating at wavelength from 2.5 - 240 μm. ISO was launched in November 1995 by an Ariane 4 launcher, and operated successfully for 30 months, well above the requirement of 18 months.

Many software packages and systems have been developed by ESA and the PI consortia to perform the standard product generation (the 'pipeline processing') and to calibrate the instruments and perform interactive data analysis.

The pipeline is basically FORTRAN based and runs under VMS, while the interactive analysis systems are built upon the commercial Interactive Data Language (IDL), distributed by Research Systems Inc, and run under various operating systems.

The pipeline products are available in the ISO data archive at http://www.iso.vilspa.esa.es/ida/, and the ISO interactive data analysis software is accessible at http://www.iso.vilspa.esa.es/archive/software/.

[1] With many thanks to:
C. Arviset, S. Guest, A. Harwood, M. Kessler, D. Lutz, A. Pollock, T. Prusti, P. Richards, J. Sternberg, B. Swinyard & E. Wieprecht

2. RÔLES OF PIPELINE AND INTERACTIVE ANALYSIS

2.1. Rôles within the project

Pipeline and interactive analysis have different rôles, both necessary to fulfil the functional tasks within the project: Interactive Analysis is the essential tool for

- analysing the instrument performance
- calibration analysis
- algorithmic development
- investigation of instrument anomalies

However, as any interactive data analysis work is rather time intensive — procedures have to be tuned to handle one or a few similar observations the best way — only a fraction of observations can be treated in detail.[2]

Therefore a pipeline, analysing *all* observations is essential in order to

- safeguard against instrumental or pointing problems
- perform quality checks whether the observation was performed successfully
- populate a database, containing observation parameters and trend analysis tables

Furthermore, as pipeline products will normally be processed to different levels, the generated products will provide convenient entry points for subsequent interactive analysis, thereby saving time by avoiding routine tasks, permitting the astronomer to concentrate on his expert job.

2.2. Different user communities

Each successful project will have a user community not directly involved with the project. For ISO, this user community can be split into three main groups:

instrument experts (active instrument data users)
They are knowledgeable of the interactive analysis packages, well aware of instrumental problems and data reduction pitfalls. They perform finely tuned data analysis which results in excellently reduced observations

[2] Even during ISO's performance verification phase, less than 30% of all calibration could be inspected in detail (Metcalfe L 2001)

community astronomers (multi-mission data users) They are infrequent users of ISO data, who can not invest considerable time or funds to get acquainted with or buying[3] complex data reduction systems of many instruments or missions. They require

- reliably and consistently reduced products, conforming to accepted standards
- easily usable final products: spectra, images, photometry, with the associated astrometric information
- readily accessible information to enable cross-identification and data mining
- catalogues containing sources and spectra

These information are ideally embedded into multi-wavelength virtual observatories

major programmes and big consortia Their expertise and requirements lie between those of instrument expert and general community astronomer

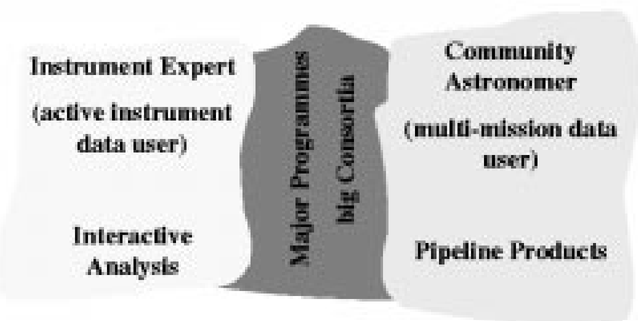

Figure 1. Relations between the three main user communities of an observatory mission. Note that the drawing is not to scale — for a successful project the number of community astronomers will be magnitudes higher than the number of instrument experts

3. ISO's DOWNLINK DEVELOPMENT APPROACH

The responsibilities for downlink software[4] were split in the following way between ESA and the PI teams:
ESA was responsible for:

- Archive development
- Database management
- Extraction from raw telemetry
- Generation of pointing files
- Overall configuration control
- Integration of delivered modules
- Standard product generation, overall quality checking and distribution of products by FTP and mailing of CDs
- Data access layer ("MC routines")
- A WWW based system to enter and track Software Problem Reports (SPR) and Software CHange Requests and Extra Wishes (SCREW)

The instrument teams were responsible for:

- Instrument calibration
- Interactive Analysis development
- Initial specification, design and development of the instrument-specific software
- Development, verification and delivery of improvements to the pipeline code
- Delivery of calibration files for the pipeline
- Detailed checking of observations with questionable quality
- Supporting the initial ground segment integration and testing
- Scientific validation of the pipeline
- Development of the real-time instrument monitoring system

In principle this was a successful approach, and worked well during the ten years of ISO's downlink development.

4. LESSONS LEARNED FROM ISO's APPROACH

This section summarises the most significant lessons learned:

4.1. Technical issues

1. Benefit from standard packages.
   Instead of using the standard FITSIO package (Pence W 1999), a new data access package (the "Modules de Coordination" or "MC routines") was written for the pipeline. Its functionality was inspired by FITSIO (from which a subset of low-level header and conversion routines were extracted) and tuned for a slightly better performance under VMS. This approach had the following drawbacks:
   - later FITSIO developments were not included
   - the routines covered only low level data access. Instrument teams had to write their own higher level access routines, leading to duplication of effort
   - it forced the pipeline development to remain on VMS
   - it produced files of a non-standard format, so own manipulation packages had to be written
2. Agree on a common product philosophy early in the mission.
   The basic data analysis steps for each processing level should be agreed on and harmonised for each instrument. Also the product layout (structure and header keywords) should reflect the common approach
3. Anticipate the need for interactive tools for users, and make them available as early in the mission as practically possible

[3] IDL is a commercial product, licenses have to be bought

[4] Downlink denotes all activities necessary to process the telemetry of the spacecraft; from the reception of the spacecraft's data, over monitoring the health of the instruments to the distribution of generated products. Its uplink counterpart are all tasks necessary to command the spacecraft and its instruments: From proposal capture to generation of the command sequences

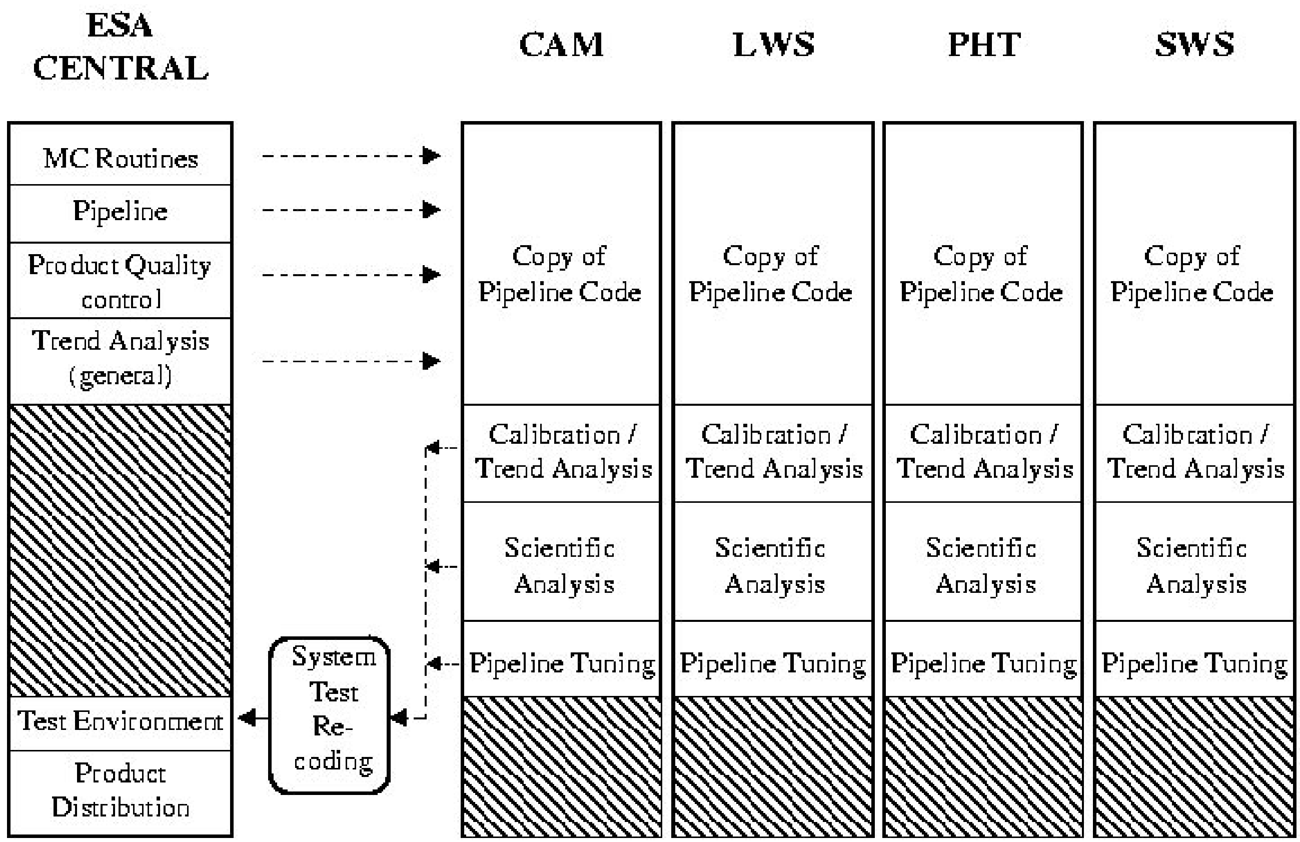

Figure 2. ISO's Pipeline Development Approach: "The Five Pillars of Wisdom"

4. Anticipate software distribution to various development sites.
   The use of a simple FTP based mirror (between computers with the same operating system) or a platform-independent configuration control and mirroring system like CoCo (Huygen R et al. 2001) will facilitate the software distribution significantly and ensure that everyone works with the same baseline
5. Ensure that all development sites have the same hardware and software configuration
   - identical processor, amount of physical memory and swap space
   - identical version of operating system and compilers
   - identical system parameters and compiler flags

   If not, you might encounter a situation that a problem at the production site can't be reproduced and understood by the development site
6. Anticipate data distribution and remote running of interactive analysis sessions between the various development sites, and provide sufficient bandwidth to permit this
7. Employ robust and telemetry driven algorithms to control essential processing steps and the generation of database tables. Do not rely on auxiliary data generated from the planned timeline of observations or on flags to be set by a user
8. Treat the pointing system as an instrument of its own. Reconstruction of pointing data can be a challenging task. To facilitate this, one has to ensure close collaboration with the pointing specialists (industry and flight dynamics)(Pollock A 2001)
9. Develop early a simulator which is able to produce realistic telemetry data, also for the more complex observing modes (e.g. scans, mosaics).
   Its output should

   - be identical to the final format
   - contain representative pointing information
   - hold representative data from instrument tests in the science blocks
   - contain realistic housekeeping information

   Such data will be invaluable to test, debug, and tune the data processing chain before launch. This reduces the timespan until usable data products can be given to the community, which is specially important for cyrogenic missions with a very limited lifetime
10. Have one accurate master clock on board the satellite, and sychronise all spacecraft and instrument telemetry with this

11. Have a centralised system to hold and access all mission data in electronic form
12. Avoid a paper-based record

4.2. Managerial issues

1. Set up management structures to encourage and reward community related efforts.
   In a set-up where the instrument teams are pressed to reduce their own observations but also have the rôle to provide calibration and data analysis tools for the benefit of the community, it is worthwhile to think about incentives (e.g. more observing time) to avoid community related efforts relegated to second-order priority
2. Plan for the whole mission. Ensure a seamless transition between the various mission phases
3. Co-location, workshops and meetings are essential.
   There is nothing like face-to-face interaction to resolve misunderstandings common in distributed projects. Therefore sufficient funds have to be available to permit these activities
4. Provide visibility and voice for all collaborating partners.
   ISO's choice of a Configuration Control Board (CCB), where software changes and schedules were discussed, facilitated this. Members could be present in person or via telecon
5. Be flexible and minimise the turn-around cycle.
   This is especially important in the early operational phase of the mission, as debugging of the pipeline code and a rapid increase in the understanding of the instruments necessitate frequent updates

4.3. Generic lessons

1. Computers are cheap, manpower is not.
   Do not accept software requirements to write a better operating system. It is cheaper and safer to purchase hardware upgrades than to spend several man-month of software engineering efforts trying to overcome the limitations of the available computer system
2. Write only technical documents which aid cohesion and mutual understanding.
   The following short-list contains the documents we consider the most useful:
   – user requirements document
   – architectural design document (holding the function calls and the product description)
   – data user manual (describing algorithms, uncertainties and caveats)
   – interface control documents
   – installation procedures
   – test-plan and -procedures
3. Don't waste the valuable time of experts with routine tasks. Try to automatise as many standard tasks as possible

5. COST-EFFICIENT DOWNLINK DEVELOPMENT

For a cost-efficient downlink development we propose the following roadmap:

1. Reuse the same code in pipeline and interactive analysis (and programs monitoring the health of the instruments in real-time) from the very start of the project.[5]
   In practice, this has the following benefits and consequences:
   – The whole system design has to be centred around the principle of re-use.
     Forbid "semi-private" packages or extensions if they
       – violate the fundamental rules of the project's coding standards
       – can't (e.g. for technical or political reasons) be readily incorporated if the need arises,
     even if it is claimed that these routines are necessary to fulfill "special" needs requested by "few" persons and will always remain outside of the "official" environment.
     If not, you face the following consequences:
       – manpower and attention is split (and might well focus on the private extensions instead of functional duties)
       – as the news of allegedly better, but not commonly available algorithms, will spread major parts, of the community will feel excluded, putting the project into a bad light
       – if the functionality of the private extensions is added to the general system, resources will still be drawn to maintain the private extensions to keep its local user community content
   – Management structures have to foster a joint development environment for all software sub-systems
   – Duplication of work like fixing bugs twice and porting improvements between pipeline and interactive analysis will be avoided
   – The larger user base gives a better chance to detect problems
   – Interactive Analysis has to be able to run in a pipeline or batch mode
2. Capture the observational goal of the astronomer while the proposal is entered, and use it to tune the pipeline processing
3. Capture common software requirements between the instruments early in the mission, and provide a common package fulfilling these needs
4. As a pre-condition to award a significant fraction of observing time to major consortia, request them to follow the project's coding standards and to collaborate with the developers of the data analysis systems.

[5] During the course of the mission, SWS started to adopt this approach and changed their pipeline from a pure FORTRAN77 pipeline to pure IDL, embedded in the overall FORTRAN environment (Wieprecht E et al. 2001)

All parties would benefit from such an arrangement:

- The consortia from the expertise of the instrument and pointing experts
- The instrument teams from the additional expertise and manpower
- The community, as the developed algorithms will be fed back into the generally available data analysis system, and high quality products, generated by data processing tuned to the special type of observations, will be fed back and made generally accessible via the project's data archive after their proprietary period has been respected

6. CONCLUSIONS

1. Before, software was seen as a consumable, and hardware was regarded as investment. This should be reversed now, with software being the precious investment
2. If you want your mission to be a success with the scientific community at large (your customers), then all community related activities (software, calibration, archive, documentation and helpdesk) are critical, and a sufficient budget for these has to be provided
3. The aim of a mission has to be to provide directly publishable products
4. Involve practitioners from the community to get user feedback. Invite them to formal reviews and listen to their recommendations
5. Increase the coupling between developers and end-users also on the informal level. Data work-shops are a good vehicle to achieve this
6. Balance the gains of basing interactive analysis systems on a commercial product against the costs levied on the astronomical community who have to pay license fees to use the interactive analysis system at their institutes
7. Balance the gains of basing data analysis systems on a commercial product which is not regulated by an agreed programming standard[6] against the danger that language changes from one version to the next are not backward compatible, and your programs have to be modified

ACKNOWLEDGEMENTS

During the history of ISO, many engineers and scientists have contributed to its software development. They are too numerous to name individually, but without their efforts, expertise, enthusiasm, dedication and professionalism the success of ISO would not have been possible.

REFERENCES

Arviset C. & Prusti T., 1999, The ISO Data Archive, ESA Bulletin number 98, 133

Burgdorf M. et al., 1998, Automatic data processing and quality control: experiences from ISO-LWS, SPIE Proceedings Vol. 3349, 47

Cutri R., 2001, Pipeline Data Processing for the Two Micron All Sky Survey: 25 TB and Counting, ADASS X, ASP Conference Series

Gabriel C. et al., 1997, The ISOPHOT Interactive Analysis PIA, a calibration and scientific analysis tool, ADASS VI, ASP Conference Series, Vol. 125, 108

Gabriel C., 2001, Development and distribution of data analysis tools: the ISOPHOT Interactive Analysis case, ESA-SP-481, Conference Proceedings "The Calibration Legacy of the ISO Mission", this issue

Hutchinson G. et al., 2001, The Legacy LIA Package, ESA-SP-481, Conference Proceedings "The Calibration Legacy of the ISO Mission", this issue

Huygen R. et al., 2001, SWS CoCo : Lessons learned about distributed Software Configuration Control, ESA-SP-481, Conference Proceedings "The Calibration Legacy of the ISO Mission", this issue

Kessler M. et al., 1996, The Infrared Space Observatory, A&A 315, L27

Metcalfe L., 2001, The ISOCAM Calibration Strategy, ESA-SP-481, Conference Proceedings "The Calibration Legacy of the ISO Mission", this issue

Ott S. et al., 1997, Design and Implementation of CIA, the ISOCAM Interactive Analysis System, ADASS VI, ASP Conference Series, Vol. 125, 34

Ott S., Gastaud R., et al., 2001, The ISOCAM Interactive Analysis System CIA: A Review of Seven Years of Development, ESA-SP-481, Conference Proceedings "The Calibration Legacy of the ISO Mission", this issue

Pence W.D., 1999, CFITSIO, v2.0" A New Full-Featured Data Interface, ADASS VIII, ASP Conference Series, Vol. 172, 487

Pollock A.M.T, 2001, The Treatment of Pointing Data in the ISO Pipeline, ESA-SP-481, Conference Proceedings "The Calibration Legacy of the ISO Mission", this issue

Pollock A.M.T. et al., 2001, The CAM Automatic Data Analysis Pipeline, ESA-SP-481, Conference Proceedings "The Calibration Legacy of the ISO Mission", this issue

Richards P. et al., 2001, The ISOPHOT Pipeline Data Processing, ESA-SP-481, Conference Proceedings "The Calibration Legacy of the ISO Mission", this issue

Sturm E., Bauer O.H., Lutz D., Wieprecht E., Wiezorrek E., et al., 1998, The ISO Spectral Analysis Package, ADASS VII, ASP Conference Series, Vol. 145

Salama A., 2001, Spacecraft: Telescope and AOCS (Pointing), ESA-SP-481, Conference Proceedings "The Calibration Legacy of the ISO Mission", this issue

Szalary A., 2001, National Virtual Observatory, ADASS X, ASP Conference Series

Wieprecht E., Lahuis F., et al., 1998, The Interaction of the ISO-SWS Pipeline software and the ISO-SWS Interactive Analysis System, ADASS VII, ASP Conference Series, Vol. 145

Wieprecht E. et al., 2001, The ISO SWS Data Analysis Software Systems, ESA-SP-481, Conference Proceedings "The Calibration Legacy of the ISO Mission", this issue

[6] contrary to C or FORTRAN, which are regulated by IEEE standards