GLAST/MPE/Mayer-Hasselwander/2002/June/04
Keys: Science, Analysis, Software, Requirement, User, Job control, Messaging, Language, Program, Visualization

Some User Requirements for Science Analysis Software

Input to 'GLAST LAT Science Tools Meeting', June 2002, SLAC

H. A. Mayer-Hasselwander

Introduction

The history of successful high-energy gamma-ray-astronomy missions extends already through 30 years from SAS-2 through COS-B to EGRET. All three missions involved rather similar instrument concepts with a tracking device as the central unit, requiring reconstruction of the individual gamma-ray events to finally obtain their incidence direction and energy. The calculation of sky exposure, the determination of the barycentric event time and the production of a skymap were the other basic processing task. The software for these ‚basic processing' tasks at the epoch of COS-B launch was ready only in rather rudimentary form, while, at the epoch of EGRET launch indeed existed in a form ready to use.
However, the ‚science analysis tools' at the launch of EGRET were still far from being useful, needed several years of further development and never reached a level, which, without a learning phase of several month (at an EGRET site), would have enabled an external user for productive usage. The reasons for this situation were basically insufficient financial provisions, not allowing for sufficient professional support (coordination, programming, version control, quality control) of the distributed effort of team members and guests who straggled to put software together which at the end indeed produced some output. So, from the evident problems of EGRET a lot can be learned, and, no doubt, the necessity to fully coordinate and produce in a professional manner not only the ‚basic processing software', but also the ‚high level science analysis tools', is now-a-days understood, requested and supported by NASA and by the GLAST management.

Status and achievements during pre-launch epoch of GLAST

Software technology and tools have enormously evolved allowing new solutions
Professional software engineering is established in the GLAST software core groups
A suite of science analysis concepts is well known from EGRET analysis
Various new science analysis concepts appear of interest and need exploration
Expected worldwide usage of the data requires a fully tested and documented science software suite, easy to handle and complete, and publicly available already before launch
Well defined final and intermediate data structures and databases exist
Tune-able Diffuse BG model exists
Instrument response database and access exists
Sky simulation software exists
Science software suite covers ‚standard' analysis cases
Science analysis software is ‚open source'

Post-launch period Requirements

There are about three different phases of software usage:

1) the development phase using test data (mainly pre-launch)
2) the scientific test and modification phase using real data (continuing throughout mission)
3) the routine production phase.

In phase 1) the user interface usually is still not well developed and of lower priority. Setting up the test jobs usually does not require comfortable parameter input concepts.
In phase 2) a GUI usually is available and helpful for setting up the jobs, comfortable ergonomic parameter input is still not a high priority necessity. However, a concept for storing and accessing parameter datasets is preferably already implemented.
For phase 3) an ergonomic elegant job setup concept is essential because many parameters usually are involved and many datasets might be addressed. (It is absolutely boring if masks are presented for job parameter input where identical parameters have to be put in repeatedly or in unpractical units and so on.)
The point is here, that this requirement is not directly experienced by the developers and therefore explicitly needs to be formulated and emphasized in the requirement documents.

In phase 3) it is also usual that automatic sequencing of various analysis steps needs to be foreseen and has to allow for unforeseen combinations, so input needs to offer relevant flexibility in a rather general manner.
One of the typical task scenarios could look like:

- Select event datasets
- Loop over a set of energy ranges
- Produce a set of region maps
- Analyze and subtract known sources
- Search for residual sources
- Derive best source positions
- Loop over energy ranges again
- Analyze residual sources at best positions
- Plot residual maps
- Plot source position error contours
- Derive ‚true' source spectra
- Parameterize spectra
- Plot source spectra
- Make catalog entries for new sources

In view of the rich analysis experience from EGRET and also in view of thousands of sources to be detected and regularly monitored the basic cataloging task will turn out to be a routine operation, not involving any ‚science'. So the new science challenge starts beyond cataloging and, as in the past, will require writing new additional software beyond the basic suite. This defines the requirement of making it easy for the research scientist to adapt existing standard software by modification or addition of modules for the particular task he envisages. From this situation the following requirements emerge.

Programming Languages

Only a minimum set of programming languages may be used for high level science analysis software e.g.

- C++, ROOT, ROOT as script (all C++ based) **or**
- C++, IDL, PERL
- C++, IDL, XML

(It was the most awful experience of EGRET that uncontrolled a multitude of languages (Fortran, C, C++, various UNIX shell scripts, IDL , ?) was used and mixed up even for a single analysis task.
The above given examples might not be realistic, so it might be appropriate to conduct a search for a satisfying language suite. Particularly discouraging is the use of scripts to control program flow. Program flow to a large extent can better be controlled by sequential use of parameter sets (e.g. in XML) in the job control dataset.
A restrictive decision should be made here very early (actually now) by the management.
Only if control is applied here the ‚science analysis software suite' may grow in favor of the community, avoiding duplication of work, making life easier for everybody and optimizing the scientific output of the mission at the end. At some point in EGRET software life cycle further development became nearly impossible because the existing suite was too inhomogeneous and too difficult to understand to implement extensions.

Computing Platforms

WEB based computing, using a central facility, very likely will be one way to work (currently it is not practical) in not to far future. However, probably conventional local platforms will also be used throughout the mission; the number of supported operating systems need to be limited to 2 or 3 (WIN, Linux, Solaris) to limit support effort.

Job Control dataset concept

The usual concept is suggested where:

- In case of a first start a default test parameter dataset is automatically loaded
- In case of a repeated start the previously used parameter dataset is loaded
- If a particular existing parameter dataset is to be used, then its name can be specified.
    - It should be possible to modify an existing parameter dataset using an editor.
    - It might be interesting to foresee also a possibility to modify existing parameter datasets through a GUI.

Naming the output datasets should be done automatically by modifying a single name provided on input.

It is the experience of the past missions that it is not unusual that analysis tasks for varying reasons need to be repeated with more or less modified parameter sets. Thus easy modification capability of already existing job control parameter data sets is a requirement. Facilities should exist which allow to print or display job control datasets in a well formatted manner for easy checking correctness and for documenting the processing task.

Messaging and Output

A positive experience from some EGRET programs was the concept of using separate output channels for documenting the processing progress by:
Copying the processed input parameters 1:1 to output dataset A
Writing error messages to output dataset B
Writing the processing log to dataset C (include information on program versions, date, installation)
Writing intermediate results to dataset D
Writing final results to datasets E, F, ...

Visualization

A very flexible mapping tool is required, allowing for instrumental, celestial, galactic grids in various projections (rectangular, Aithoff) with wide choice of bin and area sizes. Overlays of other maps and of source positions, error circles is needed and arrangement of several plots on a page. Scales and legends need to be customer defined and the corresponding control datasets must be storable for repeated use. Similarly, tools for other standard tasks like spectrum displays and phase histograms etc. will be needed.
It is very likely that a dedicated custom made visualization suite will be required, like for most of larger previous projects.
However, it might be worthwhile to investigate of parts of existing tools from optical, X-ray, IR projects or commercial tools (XSPEC, MIDAS, DS9, ROOT, IDL, COREL, etc) could be adapted and integrated. At least investigating them might help to learn what kind of features might be useful and elegant and thus possibly reproduced within the GLAST suite.

The effort put in streamlining the science analysis processing chains is absolutely valuable and rewarding: it will save time for hundreds of users over a decade of GLAST analysis and thus will enhance science output.