*Astronomical Data Analysis Software and Systems XIII*
*ASP Conference Series, Vol. 314, 2004*
*F. Ochsenbein, M. Allen, and D. Egret, eds.*

# The XMM-Newton SAS - Distributed Development and Maintenance of a Large Science Analysis System: A Critical Analysis

Carlos Gabriel[1], John Hoar[1], Aitor Ibarra[1], Uwe Lammers[2], Eduardo Ojero[1], Richard Saxton[1], Giuseppe Vacanti[2]

*XMM-Newton Science Operations Centre, Science Operations and Data Systems Division of ESA, European Space Agency*

Mike Denby, Duncan Fyfe, Julian Osborne

*Department of Physics and Astronomy, University of Leicester, Leicester, LE1 7RH, UK*

**Abstract.** The XMM-Newton Scientific Analysis System (SAS) is the software used for the reduction and calibration of data taken with the XMM-Newton satellite instruments leading to almost 400 refereed scientific papers published in the last 2.5 years. Its maintenance, further development and distribution is under the responsibility of the XMM-Newton Science Operations Centre together with the Survey Science Centre, representing a collaborative effort of more than 30 scientific institutes.

Developed in C++, Fortran 90/95 and Perl, the SAS makes large use of open software packages such as *ds9* for image display (SAO-R&D Software Suite), Grace, LHEASOFT and cfitsio (HEASARC project), pgplot, fftw and the non-commercial version of Qt (TrollTech).

The combination of supporting several versions of SAS for multiple platforms (including SunOS, DEC, many Linux flavours and MacOS) in a widely distributed development process which makes use of a suite of external packages and libraries presents substantial issues for the integrity of the SAS maintenance and development. A further challenge comes from the necessity of maintaining the flexibility of a software package evolving together with progress made in instrument calibration and analysis refinement, whilst at the same time being the source of all official products of the XMM-Newton mission. To cope with this requirement, a sophisticated system for continuous integration and testing on several platforms of different branches has been put in place on top of a refined development model designed for this special S/W development case.

The SAS is considered now a mature system. We present the different aspects of its development, maintenance and distribution, extracting lessons learned for present and future projects of this magnitude.

[1]VILSPA, Villafranca del Castillo, P.O.Box 50727, 28080 Madrid, Spain

[2]ESTEC, Keplerlaan 1, 2200 AG Noordwijk, The Netherlands



© Copyright 2004 Astronomical Society of the Pacific. All rights reserved.

## 1. Introduction

The XMM-Newton Scientific Analysis System (SAS) is the main tool for offline processing of data obtained from the scientific instruments on board XMM-Newton [Jansen et al. 2001]. These consist of X-ray instruments performing imaging, spectroscopy and timing and an optical/UV camera for imaging, timing and medium-resolution dispersive spectroscopy.

The SAS runs both in interactive mode, including a complete GUI system, and in scripting mode, in which input parameters of the tasks are specified on the command line. The scripting capability is used to create the XMM-Newton Pipeline (PPS) from a subset of the SAS, for generation the official scientific mission data products.

The SAS has been in continual development for around 6 years by a team of up to 30 developers, distributed around the world. Its development therefore presents the typical difficulties of distributed development. The large use of external freeware libraries and the objective of distributing the system for multiple platforms whilst taking the most user-friendly approach possible accentuates the demand for a flexible but controlled development with a continuous integration process. Specially important in this context is that the individual developer has rapid feedback on the integration of their task into the system. Several of the techniques used in the development (e.g. continuous integration and testing) are established paradigms in Extreme Programming (XP) [see e.g. Beck and Fowler 2001].

## 2. The XMM-Newton SAS general capabilities

The main purpose of the SAS is the reduction of data from all XMM-Newton scientific instruments to the level of calibrated event lists, images, spectra, source lists and detector response matrices, allowing the observer to perform astronomical analysis using those products without the need of any special knowledge of instrumental performance or calibration. At the same time the SAS provides the observer with the means to repeat the reduction process after improvements in calibration. A complete data analysis toolset is also in place for optimizing the selection criteria in order to achieve the best signal to noise tailored to the scientific case the observer is interested in.

A modular architecture allows the SAS user to go step-by-step through the entire chain of the data reduction. SAS tasks are highly parameterised, such that almost all the different data handling steps are configurable; in particular those steps which are considered essential. In this way the effect of the presence / absence / value of those steps are easily tested.

SAS "task" GUIs are created by a common graphical software mechanism based on Qt which allows easy access to all their own specific parameters. That method produces a consistent and easy-to-use look-and-feel as compared to having to use a different GUI for each "task". All tasks can be run from the command line specifying all mandatory task parameters.

The modular structure and highly parameterised nature of the SAS allows detailed configuration of the data manipulation process; when combined with the command line interface it provides a powerful and efficient data reduction

facility. It is very easy to compose data reduction scripts since the output produced by each task is potentially input to a task following logically in one of the possible data reduction chains. This is taken to its logical conclusion in the XMM-Newton Pipeline, a sophisticated processing system with the SAS at the core, developed by the SSC [Fyfe et al 2001].

The log file of any SAS data reduction session (irrespective of whether tasks are run from GUI or command line) contains information on all the tasks performed, including the parameters used, marked in a way that they are easily recognized as such by the reader. Use of this capability to record every single performed call is made by a special task, which creates automatically from such a logfile a corresponding executable script.

Access to general information on SAS, on-line documentation, binaries, special reduction scripts and much more is provided through the XMM-Newton webpages (http://xmm.vilspa.esa.es/).

## 3. A fully distributed development

A natural task distribution among the developers collaborating in the project has been established since the beginning of development. A central development group within ESA's SOC, composed of scientists and software engineers, takes care of all the infrastructure tasks, including the data and calibration access layers. A highly distributed team, coordinated by the Survey Science Center at the University of Leicester[1] and composed mainly of scientists working closely with the different XMM-Newton instrument teams, is in charge of the instruments' data processing tasks. A high level of communication is needed in such a distributed environment, which is achieved through a mailing list dedicated to development, as the forum for discussion, exchange of ideas and communication of new developments.

In addition a number of SAS Working Group meetings are organized every year, bringing the developers together to communicate and report on the status and future of each area of development. Two separate configuration control systems for Software Problem Report (SPR) and Software Change Request (SCR) were established at the XMM-Newton SOC and the SSC respectively, reflecting the "local" handling of the SAS tasks. These are visible to the entire project and automatically send emails to the task developers and managers when there is a change of report status. A Configuration Control Board (SAS CCB), composed by members of both the SOC and the SSC, handles general strategy questions, change requests, release schedules, etc.

## 4. The SAS development model

In order to make possible the SAS development, with many distributed developers working through a system of honour rather than subject to authority-wielding management of their work by a central body, a special development model had to be put in place. It is based on three central elements:

[1] http://xmmssc-www.star.le.ac.uk/

- the whole system is broken down into single packages,
- the multiple dependencies among packages are taken care of by the build system,
- a thorough unit- and system-level testing approach is realized through continuous integration and testing on several platforms.

The cycle of integration is composed of following steps:

- changed SAS task packages are uploaded to the SOC central repository where a new software "manifest" is issued daily
- builds based on the issued "manifest" are performed daily on different reference systems for specific platforms and operating systems, in the SOC and other selected sites,
- the build reports containing information on the build setup, its results and tests are published automatically by the build process onto the development central web page (http://xmm.vilspa.esa.es/sasdev/integration),
- within 24 hours developers can see the results of their uploaded changes to any SAS task code, into the complete structure of SAS for all the different platforms, operating systems and different build configurations.

A number of parallel building "tracks" are used in order to allow flexibility during development ("development track") as well as a highly configuration-controlled system, as needed during the period prior to an imminent release or for official pipeline development ("release track"). Snapshots of a certain development stage can be taken at any time. The contents of a snapshot are determined by the version numbers of the packages recorded in a "manifest", which can be changed following simple rules governing packages within a "manifest" .

Every major public SAS release is accompanied by a process of scientific validation of the software. This consists in the automatic data reduction of a pre-determined set of test data performed by the PPS, followed by a thorough scientific interactive analysis. The aims of this exercise are to establish:

- which instrumental modes are fully supported by SAS,
- which scientific products SAS can produce, and
- the level of accuracy associated with those products.

The outcome of the SAS scientific validation is summarized in a report, made available through the SAS webpages.

The SAS is distributed in binary form for easy installation on a wide range of platforms and operating systems. The officially supported systems vary slightly from version to version of the SAS, following the general evolution of the target environments, in particular in the Linux sector. The platforms used currently for SAS integration are: Solaris 2.6 and 2.8, Linux Red Hat 9.0 and SuSe 7.3, Mac OS Darwin 6.6 as well as DEC Tru64 OSF5.1. These platforms are planned to be supported officially by the next release (SAS 6.0) in early 2004.

## 5. Lessons learned

The concept behind the SAS development has proven to be generally successful in many aspects, underlined by the fact that up to date almost all 400 refereed scientific papers on XMM-Newton data have been possible also due to the SAS data reduction capabilities. There are however also some aspects on the negative

side, which have resulted in larger manpower demands due to its development characteristics, such as

- the mix of programming languages (C++ and F90) used for coding SAS and the evolution of compiler support for these languages, which caused serious trouble in many occassions and avoidable work if only a single language would have been used. This gave a higher compiler dependency than would otherwise have been possible, but was a constraint forced on us by the skills of the pre-defined set of contributors;
- code reviews were infrequent during development. This could have improved the level of overall code quality. Code re-use was identified early on as a means of keeping total costs down. Lack of resources prevented wholesale code walkthroughs;
- the development of common utilities is extremely difficult in geographical distributed environments and can lead to code duplication. Given the ease with which the SAS infrastructure allows new tasks to be quickly made, this has to be considered a minor cost, to be set against the cost of determining the exact function of external software components;
- the strong dependency of the SAS on the evolution of external librarires and operation systems. The consequence are relatively high demands on maintenance.

The most positive aspects of the development are the following:

- core infrastructure developed in a central place and a very good work-split between software engineers and scientists;
- succesful delivery and integration concepts and procedures, including the use of web and automatic emails to report the status of the daily builds, and of the problems identified;
- defined standard structure for all SAS packages, including source, documentation, GUI parameter handling, test harnesses, dependencies, version, changelog and distribution specification;
- access and use of complex data (including calibration), through layers of abstraction enabling users / SAS developers without knowledge of underlying data structure intricacies or algorithms used for derived data,
- quick turnaround times due to the continuous integration and build processes,
- no imposition of any commercial S/W package on the end users,
- incarnation of a SAS subset as official pipeline (PPS), used for the derivation of the official products distributed to the observers and populating the XMM-Newton archive. The resulting large exposure of the software to the data led to rapid bug elimination.

## References

Beck, K., Fowler, M., 2001, "Planning Extreme Programming. XP", Addison-Wesley

Fyfe, D.J. et al., 2001, in Proceedings of the Conference "New Visions of the X-ray Universe", Noordwijk, in press.

Jansen, F. et al., 2001, A&A, 365, L1-L6