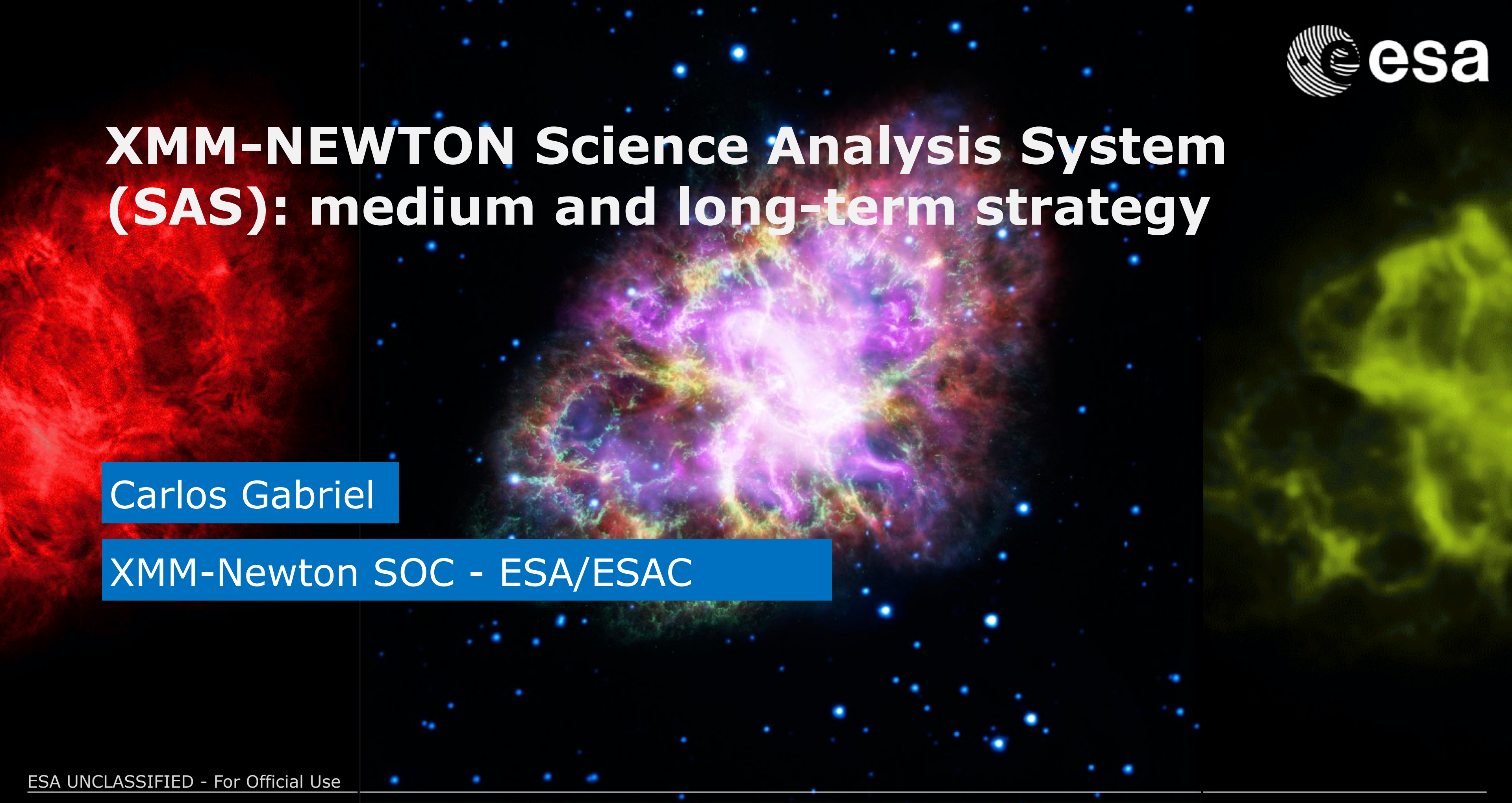


XMM-NEWTON Science Analysis System (SAS): medium and long-term strategy

Carlos Gabriel

XMM-Newton SOC - ESA/ESAC

Outline

- Science Analysis System - Requirements and Implementation
- SAS 16 - start of some changes
- Remote Interface for Science Analysis (RISA)
- Medium and Long Term Strategy
- Conclusions

Requirements for XMM-Newton's scientific analysis system

Basics:

>> **basis** for the **official** XMM-Newton products
>> **interactive** analysis system to be **used by observers**
>> capable of dealing in a **similar way** with **all the data** from all the XMM-Newton instruments

Additional requirements for an observatory's data analysis system in the XXI century:

- run on different platforms >> serving users all over the world
- run on different ways >> for more and less experienced, occasional and dedicated users, X-ray and non X-ray astronomer
- user friendly >> attracting instead of repelling
- react quickly to new developments in calibration / processing >> fundamental instrument for professional science
- free of costs >> obvious but especially important for scientists in less developed countries

>> Scientific Analysis System SAS

A key player in the 5000+ achievement?

(XMM UG: ~ "high quality tool ... instrumental in high productivity of XMM")

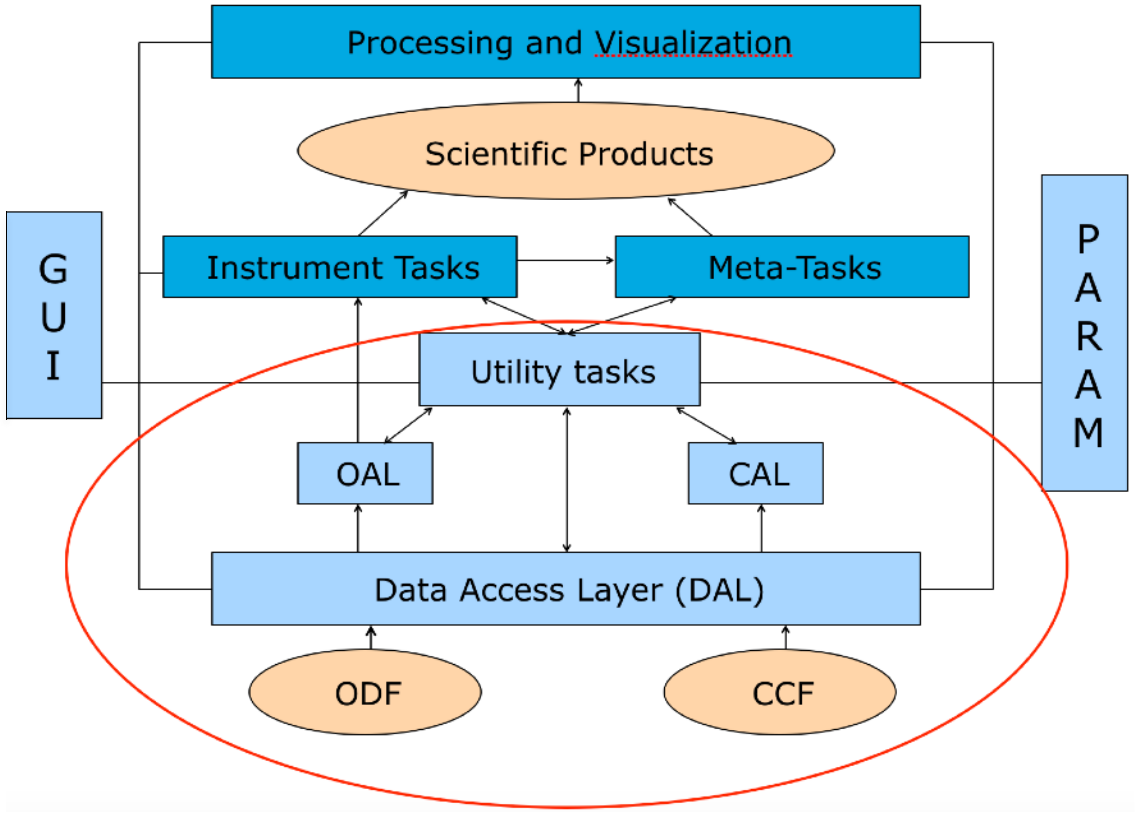
What is SAS?

- The XMM-Newton Scientific Analysis System is a **freely distributed** suite of programs (“tasks”) for dealing with data from all XMM-Newton Instruments
- **All** tasks can run from a dedicated **GUI** or from the **command line**
- **Tasks** are written in **C++** and **F90/95**
- Perl and shell scripts constitute “**metatasks**” (users can easily construct them)
- SAS compiles on: **GNU/Linux and Mac OS X** (Solaris abandoned years ago), and it is distributed in **several flavors** and as **universal VM**
- It has been **developed** by ~ **30 programmers**, working in 6 different countries, and it is **maintained** (and further developed!) by ~ **4-5 FTEs**
- A **subset of the SAS** is used as the **official pipeline** (**PPS**) for reducing the data to calibrated event lists, images, spectra, source lists (and more) distributed

    * to the PI observer ~ 2-3 weeks after the observation was performed

    * to the world via the XMM-Newton Scientific Archive (XSA) one year later

SAS subsystems scheme



Work split:

🟦 SOC

Small team with exclusive dedication in one place

🟦 SSC

Large team geographically distributed

SAS maintenance model in the last years

- Small team dealing not only with most of the SAS maintenance, but also with all of the PPS
  - only possible at the high level due to experience and excellence of team

- Distributing SAS in many binaries (32- + 64-bit versions, many Linux & Mac versions)
  - making easy its installation to the final user
  - maximising scientific return
  - … but also increasing workload on our side

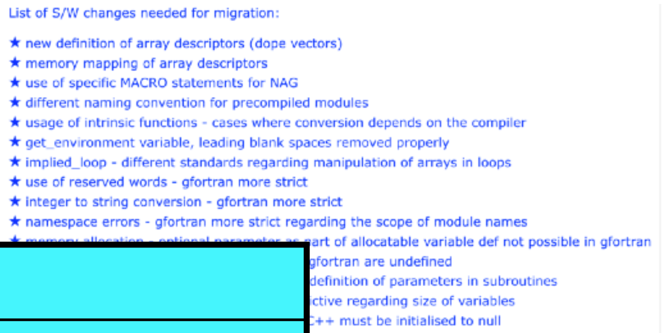
>> need to redirect efforts… first steps:

- reduce number of platforms
- simplify SAS building procedures
- start thinking in the long-term (*aka post-operational phase*)

SAS 16.0 released on 16/1/17

- SAS compiled with GNU GCC 6.2, including **gfortran 6.2** (away from NAG fortran compiler)
  Compliance with newest C++ and Fortran coding standards

  --- Main element of this release, it implied a very large effort by SAS team ---
  (a pre-requisite for future SAS compilation at user's side...)



List of S/W changes needed for migration:

★ new definition of array descriptors (dope vectors)
★ memory mapping of array descriptors
★ use of specific MACRO statements for NAG
★ different naming convention for precompiled modules
★ usage of intrinsic functions - cases where conversion depends on the compiler
★ get_environment variable, leading blank spaces removed properly
★ implied_loop - different standards regarding manipulation of arrays in loops
★ use of reserved words - gfortran more strict
★ integer to string conversion - gfortran more strict
★ namespace errors - gfortran more strict regarding the scope of module names
★ memory allocation - optional parameter as part of allocatable variable def not possible in gfortran
gfortran are undefined
definition of parameters in subroutines
ctive regarding size of variables
C++ must be initialised to null

- **No** 32bits binaries anymore + reduction to few 64bits binaries

Linux 64:

RHEL 6.8	2.6.32	2.12
Ubuntu 16.04.1LTS	4.4.0	2.23

MacOS:

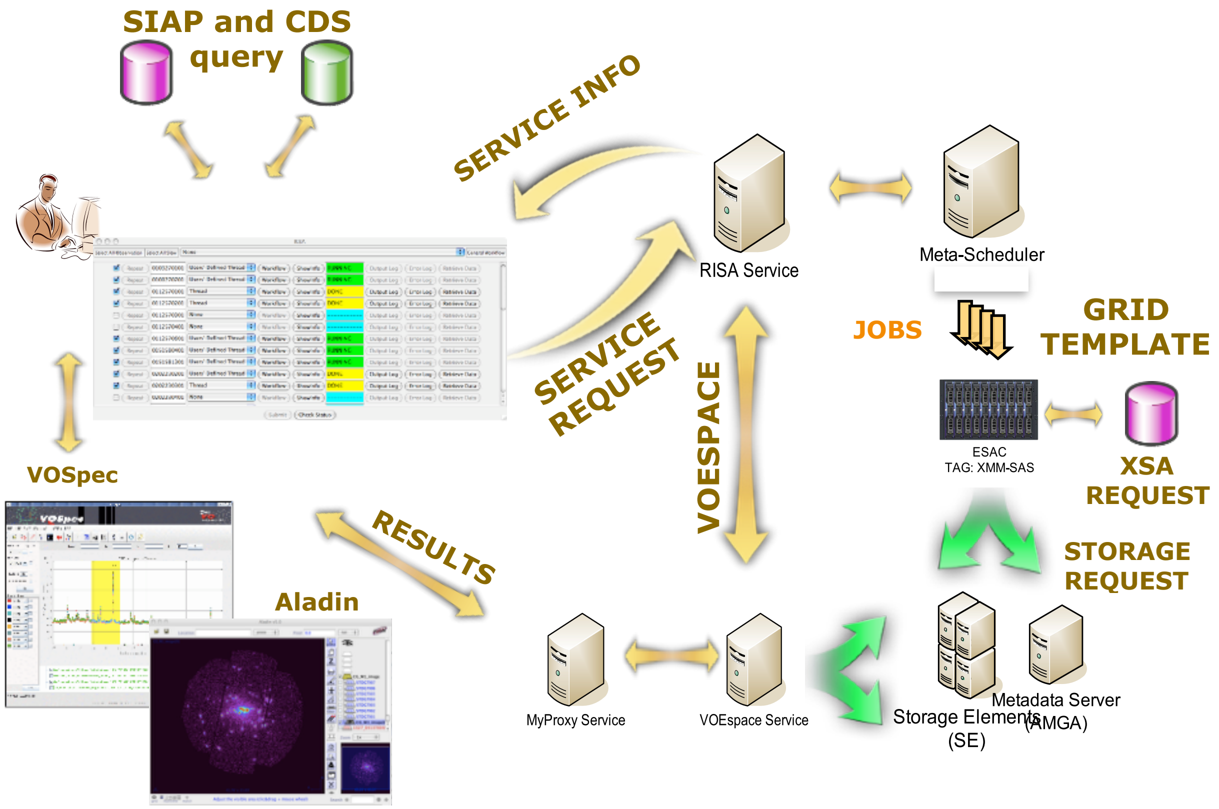
MacOS 10.10.5 (Yosemite)	Darwin 14.5.0	1213.0.0
MacOS 10.11.6 (El Capitan)	Darwin 15.6.0	1226.10.1
MacOS 10.12.3 (Sierra)	Darwin 16.3.0	1238.0.0

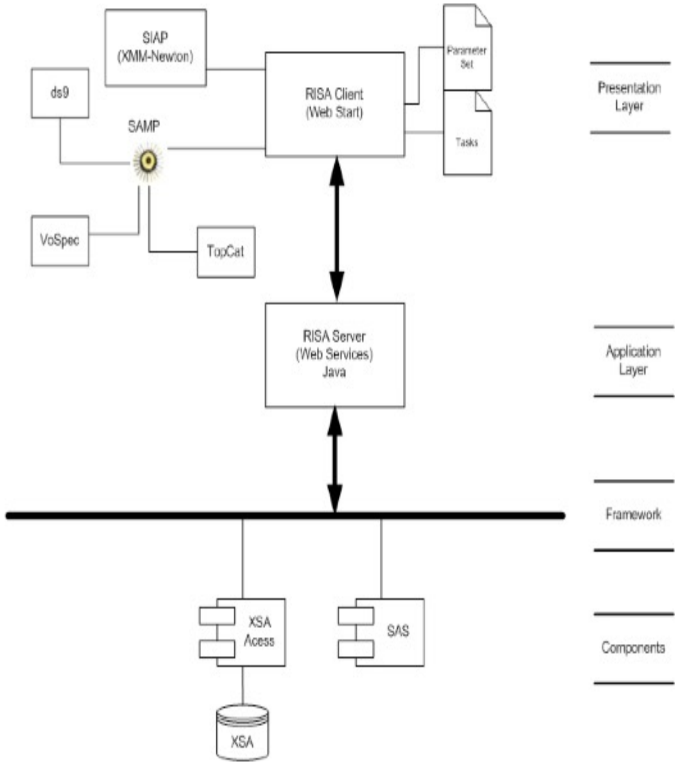
Already in SAS 15

New in SAS 16

+ 1 universal SAS-VMs (64bits) - Ubuntu16.04.1

Remote Interface for Science Analysis (RISA)





Medium- and long-term strategy

On top of "normal" maintenance (and development)... working at the same time on Post-ops

- to be better prepared if something leads to termination of XMM-Newton
- to reduce the work which will be needed for legacy during the post-operational phase (limited strictly to 2 years)

Main ideas wrt SAS data processing after EoM:

A. preserve a running SAS as long as possible
   A1) SAS Virtual Machine … Dockers *(≥ 10 years)*
   A2) RISA (Remote Interface for Science Analysis) *(5-10 years longer)*

B. give SAS code to community *(re-use (Athena…)? … further development?)*

To make possible B means: reducing complexity = modernising

B1) Compilers: maintaining close correspondence with new compilers

B2) Migration to Python in 3 areas: graphical, replacing PERL, replacing calls to HEASOFT

B3) Simplify configuration and improve documentation: making possible / easy building from source & source maintenance

Four years detailed plan (2017-2020) based on these lines

SAS VM + RISA == SAS on the net

A1 - SAS is distributed since 2006 also as Virtual Machine
Estimation: such a VM could run after EoM in the most diverse OS's for ≥ 10 years

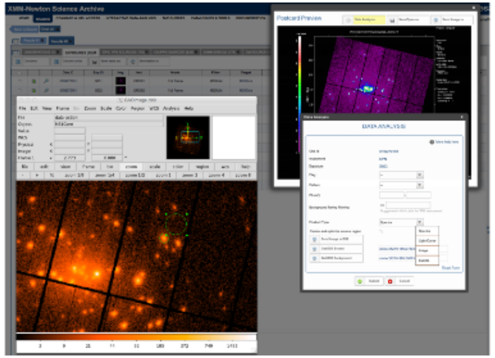
Proven: SAS-VM (2006) running today without any problem on actual OS's

A2 - RISA is a fundamental component in our long-term strategy ... it is already working though:

- on-the-fly reprocessing of archival data with latest SAS and calibration
- filtering and light data reduction services

>> integration in XSAv9.4

>> first steps in the way to a more complete / full RISA I/A service



RISA post-ops thought so far to be SAS-VM based ...
(> final SAS packed in one OS… extended life [5-10 years] in a central place)

See P11

XMM-Newton Solaris 8 operational machines (from 2004!) replaced these days…

>> RISA is an ideal system for experimenting replacement of SAS-VM by Dockers… (2018)
(Dockers would ease combination of SAS data reduction with other S/W)

Compilers, configuration and builds

B1) SAS 16: Transition to GNU provided gfortran compiler

at the same time, most modern C++ compiler version used: GCCv6.2

>> serious scrutiny of source code

>> most up-to-date standards both in Fortran 90 and in C++

Pre-requisites for providing source code for SAS compilation at user's side

• free compilers ✓

• code up-to-date with standards ✓ (… 2019 … 2021 …)

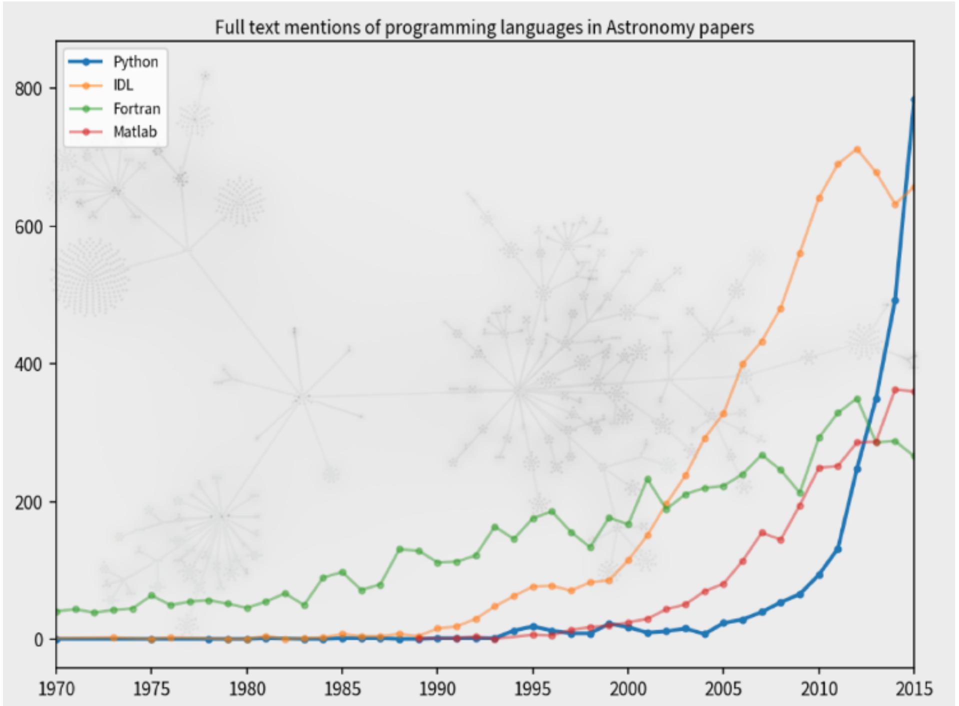
+

• simplified configuration and building procedures
  + documentation

- SAS built by SAS experts ✓
- SAS built by S/W experts (2018)
- SAS built by "normal" users (2019)

Moving areas to Python

B2 - Moving certain SAS areas to Python



Robitaille et al 2017 67% of all astronomers use Python

Why Python?

- Simplifying SAS & PPS
  >> more maintainable
- Pre-condition for a future package to be given to the community

Stepwise introduction in SAS/PPS:

t

- graphics area: first products (2017)
- graphics area: replacement of PGPLOT & Grace (2018-19)
- scripting area: replacing PERL (yes, lot of work… 41 scripts in SAS, some pretty complex) (2019)
- Heasoft area: replacing tasks depending on Heasoft (2020)

Making easier to build & maintain a running SAS

B3 - Configuration / build / documentation

Two main problems for "aliens" to deal with SAS on the source code level

- SAS is a complex piece of software >> SAS is difficult to build…

  1. replace NAG fortran compiler by gfortran ✓
  2. simplify build & configuration … extend documentation so that SAS experts can build ✓ (2017)
  3. simplify more … extend documentation so that S/W experts can build (2018)
  4. simplify even more … extend documentation so that 'aliens' can do it (2019)

- To maintain S/W written by others is difficult (don't tell us…!)

  Internal documentation is essential - Improve so that non-experts can cope with

  1. I/F type S/W (OAL, CAL, DAL) (2017-18)
  2. S/W type II (beyond calibrated event lists…) (2019)
  3. Rest of the S/W (2020)

This will be needed even before / independently of Post-ops

Summary

- SAS: maintenance efforts not decreasing, necessary for maximisation of scientific return
- Evolution: taking into account post-mission needs
- Optimising the timing for necessary structural changes, with the aim of
  - keeping analysis capabilities for decades after EoM
  - making possible maintenance of SAS code by community