ATLAS Feedback

David Quarrie

LBNL

DRQuarrie@lbl.gov

Topics

- SPI
- ROOT
- POOL, CORAL, COOL, etc.
- Geant4
- Genser
- Conclusions

SPI Feedback (1/3)

- Savannah
  - Used by ATLAS for online, offline, production
  - 49 projects
    - Developer-centric and user-centric (and some historical)
    - Developer-centric reflect CVS organization (simulation, inner detector, etc.)
    - User-centric monitored and redirected to appropriate developer-centric
      - Simplified user interface
  - Some concerns in production environment about duplicate bugs
- External Software packages
  - ATLAS uses SPI installation where available
  - We have asked for several more and SPI has been very responsive
  - Use LCGCMT glue packages
    - Shared with LHCb
    - Now share access to CVS repository
    - Some technical issues still remain (but on CMT team, not SPI)

SPI Feedback (2/3)

- Bugfix build strategy
  - ATLAS strongly supports the new build strategy
    - In order to incorporate bug-fixes we used to have to move to new development versions
    - Now dedicated bugfix versions built
      - Other layered LCG-AA products rebuilt if necessary above these
        - New version number with no code changes
- Proactive integration/validation
  - We rely heavily on nightly builds to provide "instantaneous" integration feedback
  - Requires dedicated build machines
  - Lack of an adequate number of such machines hampered our ability to try trial versions of external software (such as LCG-AA products)
  - However, now have more machines, and support the proposal to attempt more early integration and feedback
  - We think the LCG-AA should adopt nightly builds
    - Reduce project integration times
  - Note that such proactive integration can only reduce the possibility of bugs being found once the releases are built, it can't eliminate them

SPI Feedback (3/3)

- Platforms, compilers and Middleware/Fabric
  - Good interaction between the experiments
  - Still some concerns about tightness of coupling to Linux certification and grid middleware
    - We appear to be informed of decisions rather than being an integral part of making them
    - But things are a bit better than they were
  - Not possible to dictate to remote sites which versions of middleware/fabric software they install
    - Many sites are multi-purpose with ATLAS as only one of several clients
  - How to improve testing against different configurations and communication?
- Build and Configuration Tool
  - ATLAS uses CMT and has a software agreement with Orsay (developers)
    - "Guarantees" continued support and maintenance
  - Strongly support the proposal to migrate the LCG-AA projects to CMT

ROOT Feedback

- Reminder
  - ATLAS (like LHCb) uses Gaudi framework
    - Component-based; heavy use of (pure) abstract interfaces
  - ROOT initially used for I/O and end-user physics analysis
- Tighter integration of ROOT is one area where cultural differences need to be reconciled
- Another reminder that the original RTAG model was a client/provider coupling to ROOT
  - Tighter coupling was major outcome of review last year
- While ATLAS supported the migration model proposed by Pere (and described in his opening talk at this review), the reality was much more painful than expected
  - We had estimated ~2-3 months to perform migration
  - The reality was that it took ~6 months
- Several lessons to learn
- I'll go through the chronology and lessons in next slides

ROOT5 Migration Chronology

- 29 Sep 2005
  - Dedicated trial release built (10.5.1) for early tests using Reflex/ROOT4
- 16 Nov
  - Dedicated ROOT5-based nightly build (11.1.0-root5) established
- 28 Nov
  - New EDM component (DataVector) introduced that depended upon ROOT5
- 24 Jan 2006
  - Dedicated nightly build (11.2.0-root5) established
- 16 Feb
  - Release 11.3.0 built as project-based (last ROOT4 release)
- 18 Feb
  - Release 11.4.0 nightlies opened with ROOT 5
- 21 Mar
  - Release 11.4.0 built - first release based upon ROOT5

ROOT5 Migration Problems

- Took a long time to realize that ROOT5/Reflex didn't support persistency that was implemented by POOL using ROOT4/Reflection
  - No support for virtual inheritance, which was supported by POOL and which had been used by ATLAS
- This deficiency, the delay in recognizing it, and having to recover from it, was the single major contributor to the delay
- Migration predated the LCG-AA bugfix release strategy which caused some instability problems
- We introduced new capability (DataVector) too early and that obscured the real problem
- Our testing was inadequate
  - Required reconstruction to work before we could fully run EDM tests
- Simultaneous introduction of project builds muddied the waters
  - We attempted to decouple but the schedules for each of these essentially converged
- Schedule was driven by need to new COOL functionality
  - But COOL developers also spent a lot longer on this migration than expected

ROOT5 Migration Lessons

- Lessons for both ATLAS and LCG-AA!
- In hindsight several consequences of the migration were not foreseen but should have been
  - E.g. Lack of support for virtual inheritance, duplicate dictionaries
- Failures in communication/documentation
  - Vital information was known on each side but didn't get propagated across
- Try to stage migration so can compare against stable baseline
- Significant concern that impact of future migrations should not be underestimated
  - The major remaining migration is for the plugins - three in use
    - Gaudi components
    - SEAL
    - ROOT
- Since we are even closer to LHC turn-on next year we need to be absolutely sure we minimize any new disruption

ROOT Team Responsiveness

- In general the new ROOT team have been very responsive to ATLAS requests and bugfixes
- One major ATLAS request has not been adopted into the plan with timescale
  - Request made in Oct 2005 for ability to read ROOT class into differently named class as part of an extended schema evolution strategy
  - Unmet requirement from the Persistency Management RTAG Report from April 2002:
    - "These considerations imply the following requirements. - The architecture should assume that a single transient type Ti may be restorable from many different persistent representations {Pj}. - The architecture should assume that a particular persistent representation Pj may be used to initialize transient objects of various types {Ti}.The consequence of the previous is that on reading, the persistence infrastructure should not presume that the type of the transient object that will be initialized can be deduced simply by inspecting the persistent representation that will be used as input."
  - After 4.5 years still no timescale given for this
- Recently Scott Snyder discovered a partial work-around using StreamerInfo APIs
  - Would like to ensure that this API is stable and documented
  - But this might not be an optimal solution so we would still like a timescale

Other ROOT Aspects (1/2)

- ROOT I/O performance and schema evolution
  - This is an area of attracting lots of attention within ATLAS
  - Goal is to provide greater flexibility in schema evolution than is currently provided natively by ROOT
    - Transient/persistent split
    - AOD in split mode?
    - Schema evolution constraints with split mode?
    - POOL overheads vs. pure ROOT
- Should/will ROOT support a more complex persistent EDM?
  - Virtual inheritance, STL containers, etc.?
- We need to improve the communication between ATLAS & ROOT team
  - Remark is bi-directional

Other ROOT Aspects (2/2)

- Maths Libraries
  - ATLAS was an early proponent of light-weight linear algebra and physics vector libraries
  - However, migration to use e.g. smatrix classes still ongoing and slow
    - In some cases the rank of the matrix is not known at compile time
    - Heavy usage of CLHEP classes for geometry
  - Strategy is to migrate EDM classes first and then the geometry classes
  - Performance of CLHEP linear algebra classes still of concern
- Reminder that ATLAS provides primary PyROOT developer
  - Wim Lavrijsen
- Some interest has been expressed within ATLAS to explore the possibility of interfacing Gaudi/Athena based applications to PROOF
  - Needless to say we'd like to share the effort with someone else!

POOL/Reflex/Relax/CORAL

- As mentioned previously we are still concerned with POOL I/O performance even though the POOL overheads appear to be small wrt ROOT
- I don't have separate slides for the other products
- This doesn't reflect lack of interest or use within ATLAS
- We heavily use all of them, rely on them and in general they meet the needs of the experiment

COOL (1/2)

- ATLAS making extensive use of COOL 1.3 in both online and offline
  - Store calibration, alignment, online configuration and DCS data
  - Over 50GB of COOL data from online detector commissioning (and reco/analysis), offline data challenges and legacy combined testbeam data
  - More and more ATLAS software is making use of COOL
  - A success!
- We're very concerned about COOL manpower situation
  - Several medium term requests, channels data, multichannel bulk insert, payload queries, full Frontier support (almost there), etc., have been around for 6-12 months but not yet addressed
  - ATLAS lost very active developer (Sven Schmidt) so some of the shortfall is result of this
- No realistic medium term plan/schedule
  - Mainly due to manpower shortfall
  - Many deliverables have been a few months away for 6 months or more
  - COOL development significantly affected by changes in LCG Infrastructure (SEAL, ROOT)
  - Concern about further disruptions
    - E.g. SEAL plug-in manager

COOL 2/2

- Some longer term concerns about scaling of COOL schema model
  - 1 COOL folder = 1-3 COOL tables
  - Optimizations proposed by ATLAS Oracle experts, but…
    - Lack of manpower again
    - Potentially very disruptive because of radically changed schema
- Bottom line(s):
  - ATLAS heavily committed to and dependent on COOL
  - COOL team heavily hampered by lack of manpower
  - ATLAS very concerned about rate of progress

Geant4 (1/3)

- ATLAS are very active users, developers and validators for Geant4
- It's our primary simulation engine and we are heavily involved in validation
- We still maintain our own installations for flexibility and validation
- Currently Geant4.7.1 is in production
- Geant4.8 is being validated
  - Problem with doubling of cpu time per event with new multiple scattering
  - Hybrid using "old" multiple scattering being validated
- Propose to introduce hybrid Geant4.8 for our next production release (release 13), due in Jan 2007

Geant4 (2/3)

- Priorities
  - Need to ensure physics needs of LHC have high priority
    - Hadronics
    - Backgrounds
    - Geometry
  - Discrepancy between ATLAS testbeam data and Geant4 in description of longitudinal pion shower profiles still there after 2 years
- Performance
  - New multiple scattering introduced x2 performance penalty
  - Focus should be on better physics with good performance rather than good physics at all costs
- Improved geometry primitives
  - ATLAS still needs lots of tricks with memory penalties to describe its geometry adequately
  - Many of the proposed ideas have not yet resulted in concrete capabilities
    - Parallel geometries, phantom volumes, etc.

Geant4 (3/3)

- Better integration with LCG-AA software
  - E.g. Development of Boost-based Python UI rather than PyROOT+Reflex etc.
  - Persistency, histograms, etc.
- Better infrastructure/framework
  - Still difficult to integrate into ATLAS framework
  - Attempts to decouple Geant4 toolkit from framework not completed
- Bottom line(s):
  - Validation ongoing (and will never end)
  - Some concerns about cpu & memory performance priorities

Genser

- ATLAS heavily uses generators from Genser distribution
  - Essentially uses Genser version where available
- The following Generators are in use by ATLAS
  - Cascade
  - Herwig
  - Hijing
  - Isajet
  - Lhapdf
  - Pythia
- Essentially the only remaining coupling to CERNLIB
- Bottom line:
- Provides a necessary and critical service and we will continue to support it

Conclusions

- ATLAS is fully committed to LCG-AA projects
- We still provide some developer support
  - But this is getting harder to sustain as we get closer to experiment turn-on
- We are strongly supportive of moves to improve release build infrastructure
  - Bugfix builds
  - Proactive validation
  - Proposal to migrate to CMT
- We are very concerned about lack of manpower and therefore progress with COOL
- Migration to ROOT5 was significantly more painful than predicted
  - Lessons to be learned by both ATLAS and LCG-AA
- We are very concerned about possible disruption from future proposed migrations
  - SEAL/ROOT plug-ins