ATLAS Software Status and Plans

Graeme Stewart & Walter Lampl

University of Glasgow | School of Physics & Astronomy

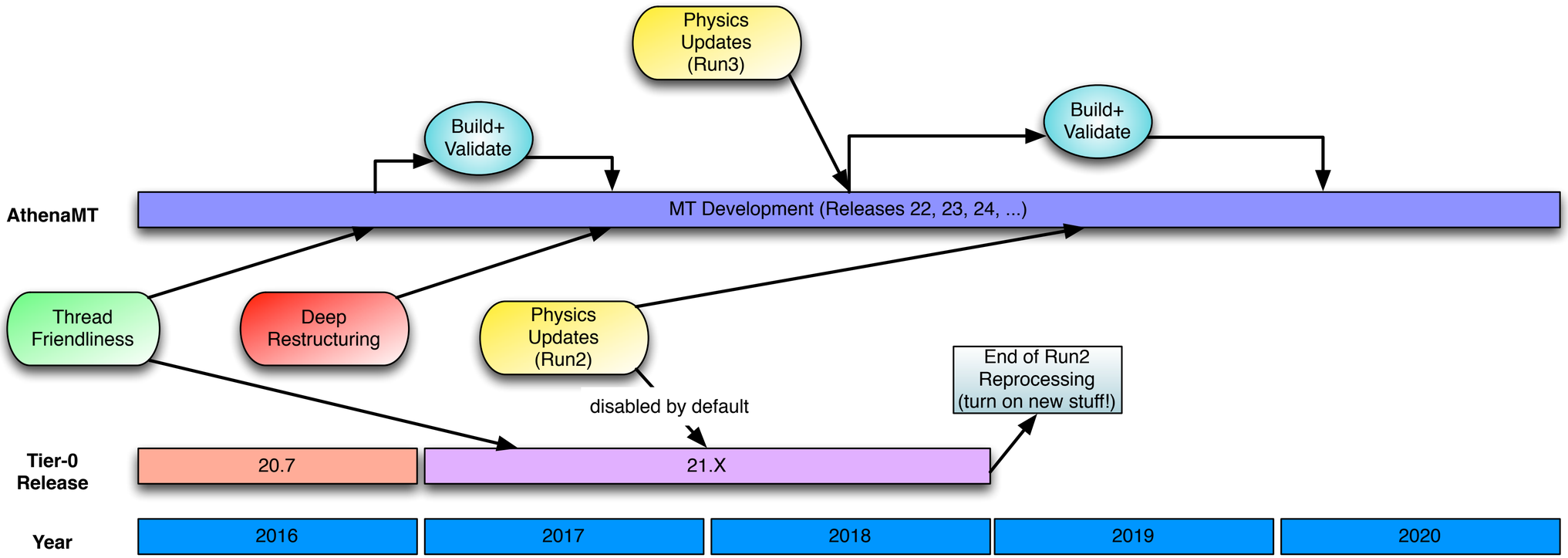
ATLAS Site Jamboree
2017-01-19

Current Software Releases

- The current major software releases are
  - 19.2 — Event generation and MC15 simulation
  - 20.7 — Last year's Tier-0 release
    - Digitisation and reconstruction of MC15 samples
    - Data and monte-carlo derivations for current Physics Conferences
  - 21.0 — Next major reconstruction release, improved physics performance
    - Tier-0 in 2017 *and* 2018 (thus to the end of Run 2)
    - Will also run MC16 simulation as well as digi-reco
    - In validation now, will be used to reprocess all Run 2 data to date
      - Default release for analyses after the summer

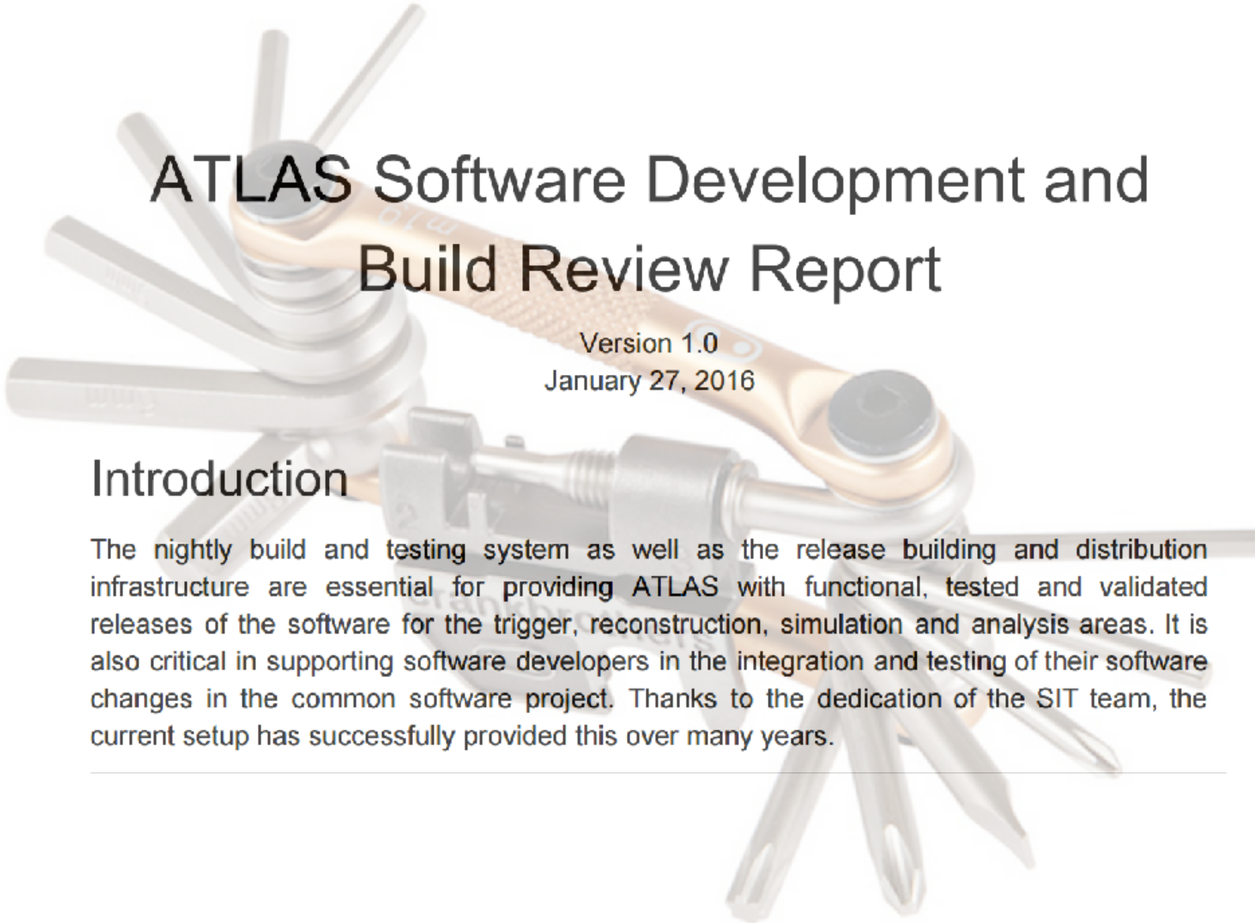
Release Updates

- Quite a complex chain of bug fixes, possible physics updates and long term improvements for multi-threading



Infrastructure

- ATLAS software infrastructure developed over many years
- Very successful
- But relied on many tools that were home made
  - Hard to maintain, hard to extend
- It was time to modernise!



CMake

- CMake is an open source cross platform build system
  - Can build, test and package software flexibly and robustly
    - Also widely used now inside of the HEP community
  - CMake had a long lead time from 2014 onwards
  - Arrived in production last year
    - All of release 21 is now built with CMake
    - Developers are happy with the new system
- Faster: build times for the full release are down to ~3 hours (fat machine)
- Scalable: CMake scales to configure the full build in under 10 minutes
  - No more projects (the ATLAS way of splitting and stacking the software build)
- Robust: incremental builds work properly
  - Enables continuous integration with fast turn around times
- Portable: builds on non-x86 hardware become much easier

Jenkins

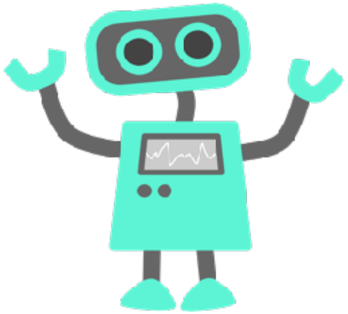
- Migration of ATLAS nightly build system (NICOS) internals to Jenkins
  - Started in December, finished this month
- Transparent to users
- Working well
- Jenkins also underpins new software development
  - Runs continuous integration on each merge request from a developer

Git

- SVN service at CERN will be shut down during LS2
  - Mainly because it's tied intimately to AFS
- Software review concluded that git was the natural migration path
- It's not a drop in replacement for SVN
  - It is *different* and *better*
- Git migration team studied the practicalities of migration
  - ATLAS software is a huge beast, not even initially clear how to structure the repository
    - 2200 Packages
    - 4M C++ lines, 1.5M python lines
    - 66k files in working copy
  - However, in the end it git scales well even to this size of repository (though AFS doesn't)
    - Any segmentation of the repository introduced complexity of the workflow and required additional tools

Development to Release Workflow

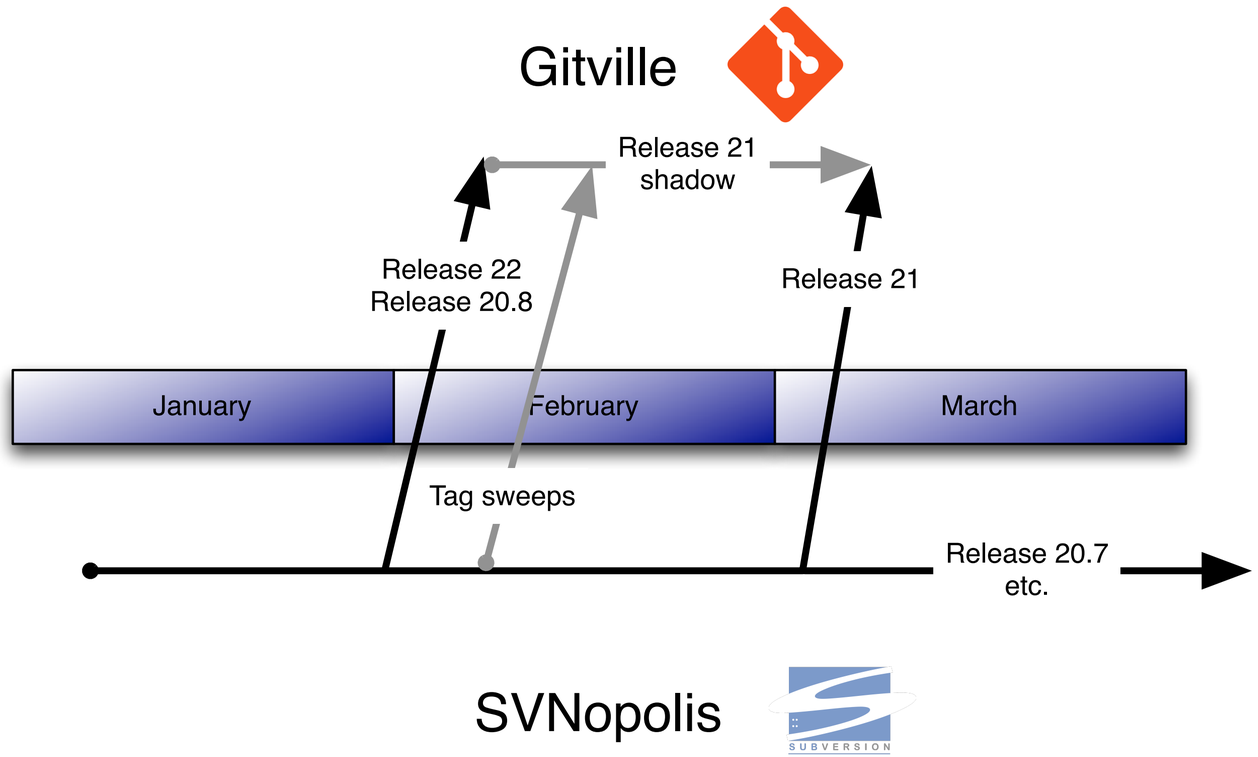
- Code development
  - See our tutorial
- Merge request handling
  - Continuous Integration and review
- Merge forwarding (~sweeps)
- Testing infrastructure
- Build and deployment procedure
  - We will deploy many more releases *directly* to CVMFS on atlas-nightlies.cern.ch
  - We will do a lot more testing on the grid in the future



AtlasBot

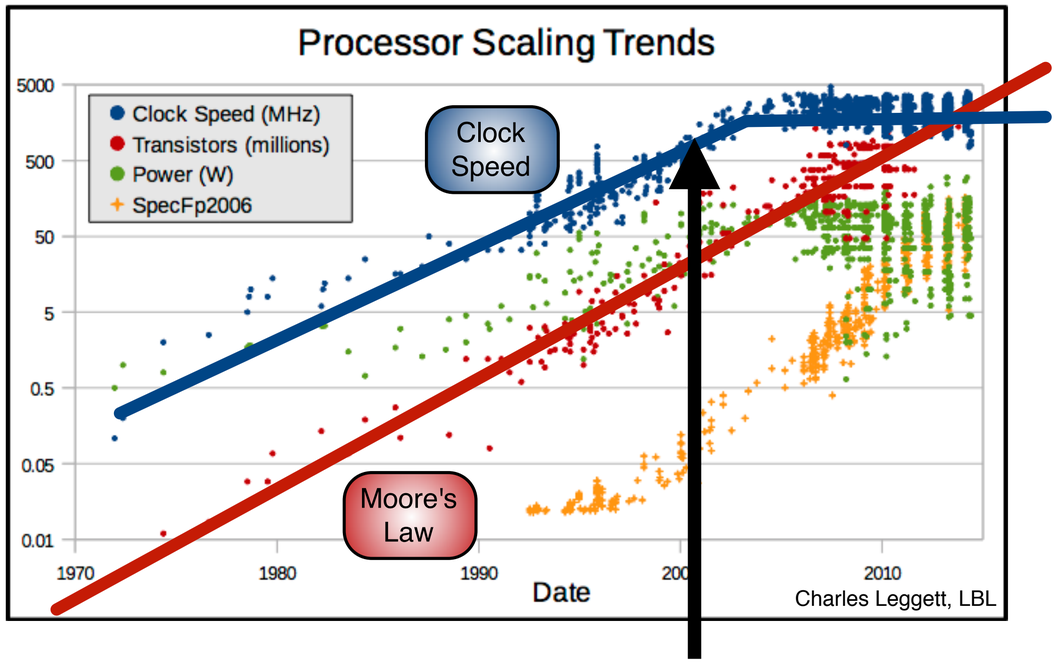
git Timescale

- Migration of development Release 22 starts in a few weeks
- Followed by Release 21
  - Avoid disturbing reprocessing
- Old releases built with CMT will not migrate into git
  - They will die off over time anyway
- This retooling will help us hugely with the major migration efforts required for Multi-Threading in Release 22



Long Term Technology Background

- Technology evolution in cpu, disk, tape and networks continues
  - However, many technologies are now mature, gains are slower and the road ahead is not entirely clear
    - e.g., Moore's Law scaling is now ~30 months and sub-10nm feature size looks prohibitively expensive
  - Certainly we are far from the halcyon days of Moore's Law scaling and the 'free lunch'
- See recent review by Bernd Panzer at ECFA workshop



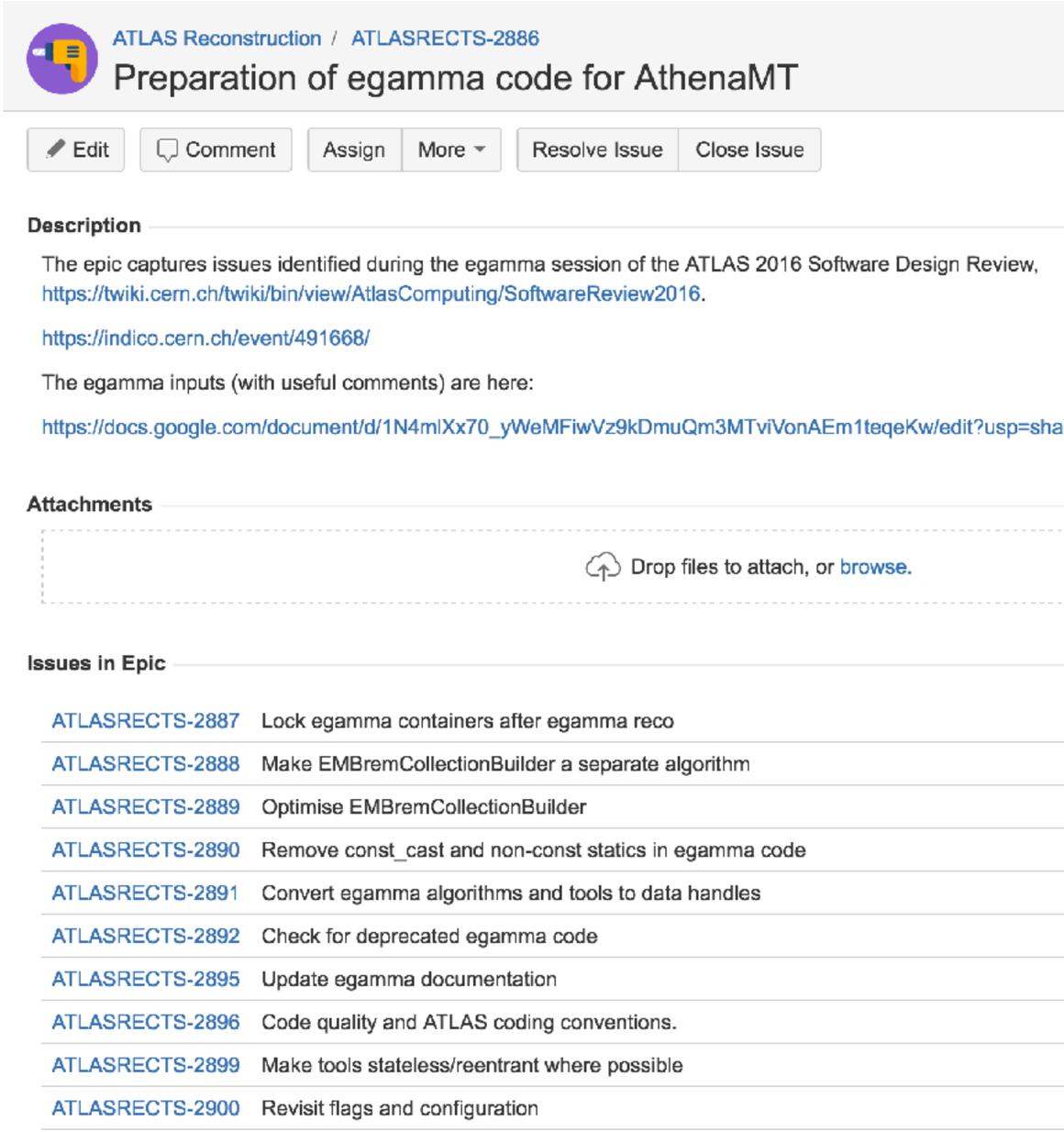
ATLAS software and computing designed here

AthenaMT

- Major Phase I upgrade for software is a *multi-threaded* version of Athena
  - In Run I we relied on serial Athena running on multiple CPU cores
  - In Run II we use multi-process Athena, AthenaMP, that shares large static memory structures like magnetic field and geometry
- However, even with AthenaMP our memory consumption is very high (4GB for full reconstruction + 1.0-1.5GB per worker process)
  - We barely fit into current grid resources and some optimised workflows, such as RAWtoALL, cannot be run on the grid
- Multi-threading shares memory very much more efficiently than multi-process
  - Not only solves a current problem, but prepares us for different architectures with less memory per core
- Need to migrate a code base of ~4 million lines of C++ (2000 packages) with many developers and a lot of history
  - General trend is for *less* developers and many of the original authors have moved on

Offline Software Reviews

- We organised a major series of software reviews last year
- Focused on the design of the code and its interaction with other algorithms and StoreGate
  - This was to give us a much more detailed idea of the code evolution necessary to deliver multi-threading
- Conclusion:
  - There is a lot to do!
  - The review process itself was an important first step
    - Relearning and documenting code that has lacked proper attention for years



Framework Progress

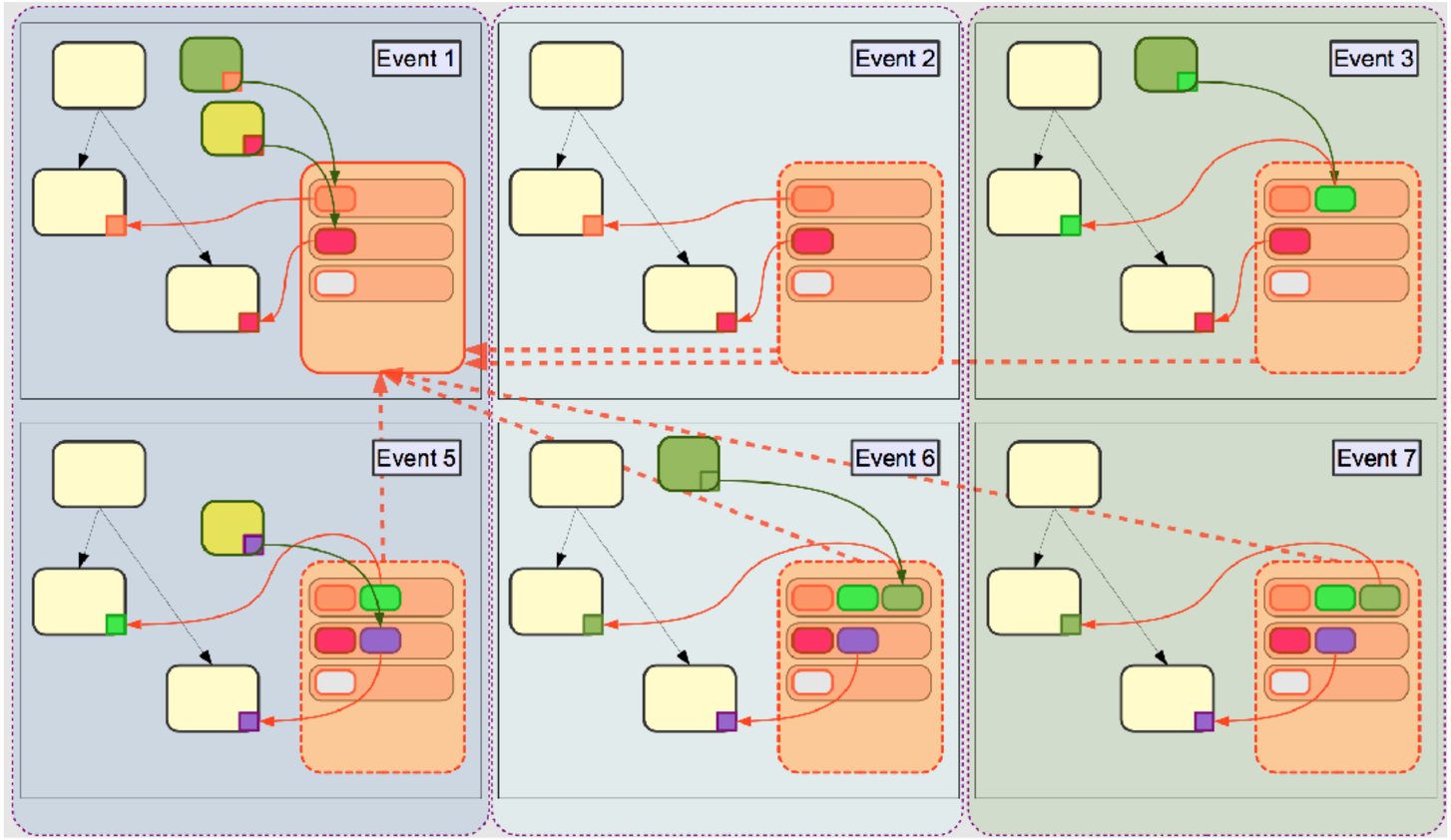
- A lot of hard work in migrating ATLAS framework code to AthenaMT is in making shared s*ervices* thread safe or able to handle multiple concurrent events.
  - Some services can be made concurrent / thread safe with simple mutexes or thread safe data structures
  - Some need more modifications to handle state information of multiple concurrent events
    - MagFieldSvc: carry event specific cache along with each request
    - THistSvc: users can choose whether to share or clone histograms
- Some need significant redesign or internal changes
  - Conditions / IOVSvc (Intervals of Validity) / GeoModelSvc (Detector Alignment)

Asynchronous Non-Event Data

- We use a lot of non-event data in our processing
  - High voltages, calibrations, alignments, etc.
  - Although a lot is quite stable, none is actually static
  - Cadence of changes can be quite high in certain circumstances
- At the moment Athena ensures that the non-event data is correct for the event being processed
  - But in a multi-threaded environment there are many events being processed at once
    - Need to make sure that they all get the correct data for their timestamp
- Generalise the idea of a data handle* to a *conditions handle*
  - Just like a data handle, but retrieves non-event data from a conditions store
  - Algorithmic client is then insensitive to the backend store

*The backend neutral way to access data, now being introduced

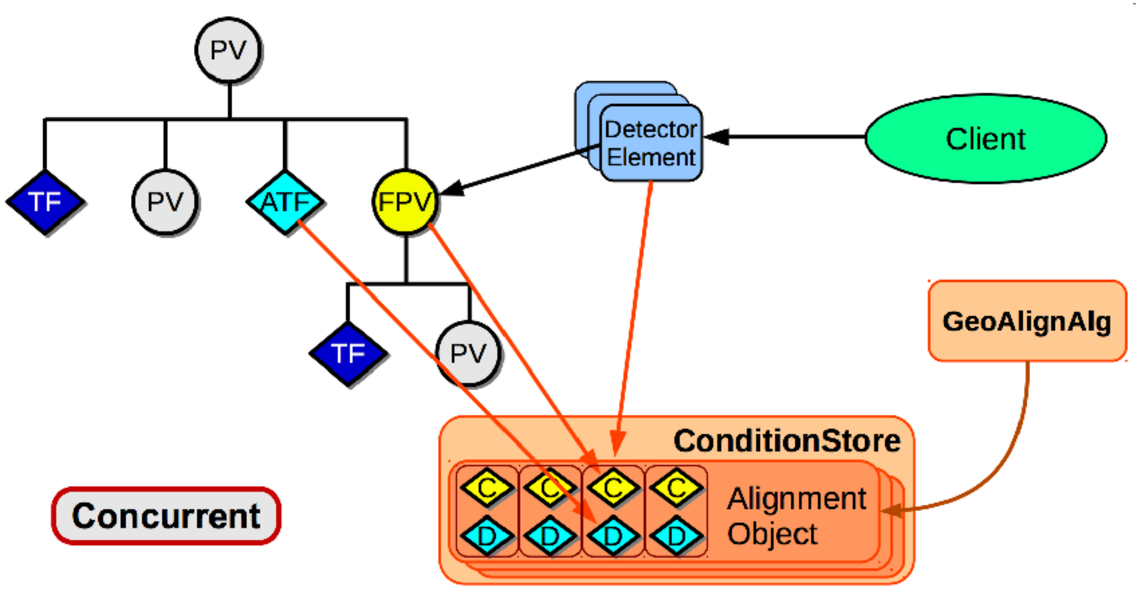
MT Conditions Store

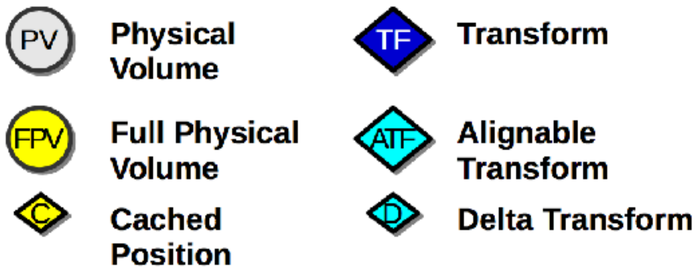


- Calibrated conditions become special algorithms run when new raw conditions are loaded
- Store must be smart to garbage collect
- N.B. Smoothing of online DCS jitters into the offline database will help a lot

Detector Geometry

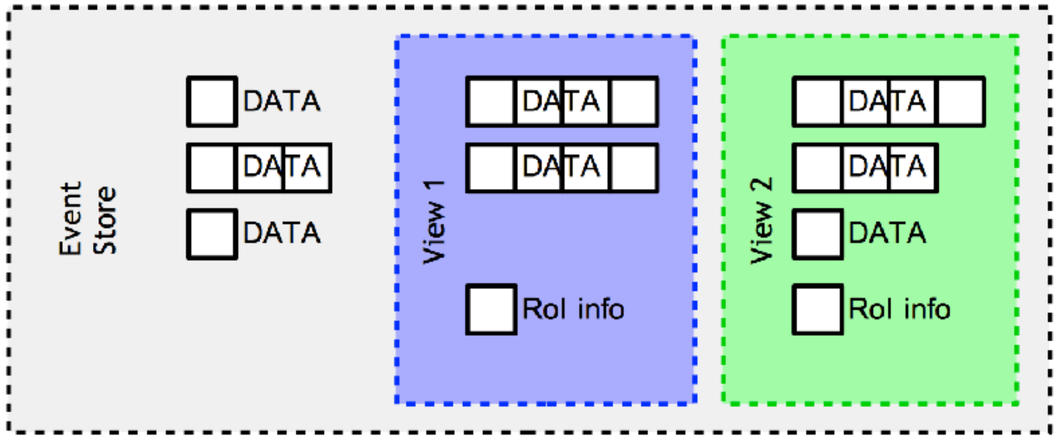
- Detector geometry is another form of non-event data that can change during processing
- Main difference is that it's huge — 100s MB
- Currently the detector elements are realigned when a new geometry IOV is encountered
- With the new GeoModelKernel the deltas are cached in the conditions store, along with the aligned elements, all per-IOV





HLT and Event Views

- Significant driver of our framework evolution is to have HLT and offline more coherent in their use of Gaudi/Athena
- Major missing component in Gaudi is the ability to process regions of interest signalled by L1 to minimise investments in rejected events
- In our updated framework regions of interest are implemented using *event views*
- All data access in the new framework is via handles
  - Hide the details of the backend store — if a view is being used it's transparent



Multi-threading in Practice

- There remains substantial development work before we can run significant amounts of a reconstruction or trigger jobs multi-threaded
- However, ATLAS simulation is a much simpler application than the above
  - ~400 software packages instead of 2200
  - Geant4 has supported multi-threading since 4.10
- In addition the multi-threading simulation strategy is to simulate one event per-thread
  - Fits with the ATLAS model where event simulation is run in a single algorithm that wraps the G4 code

Making it thread safe

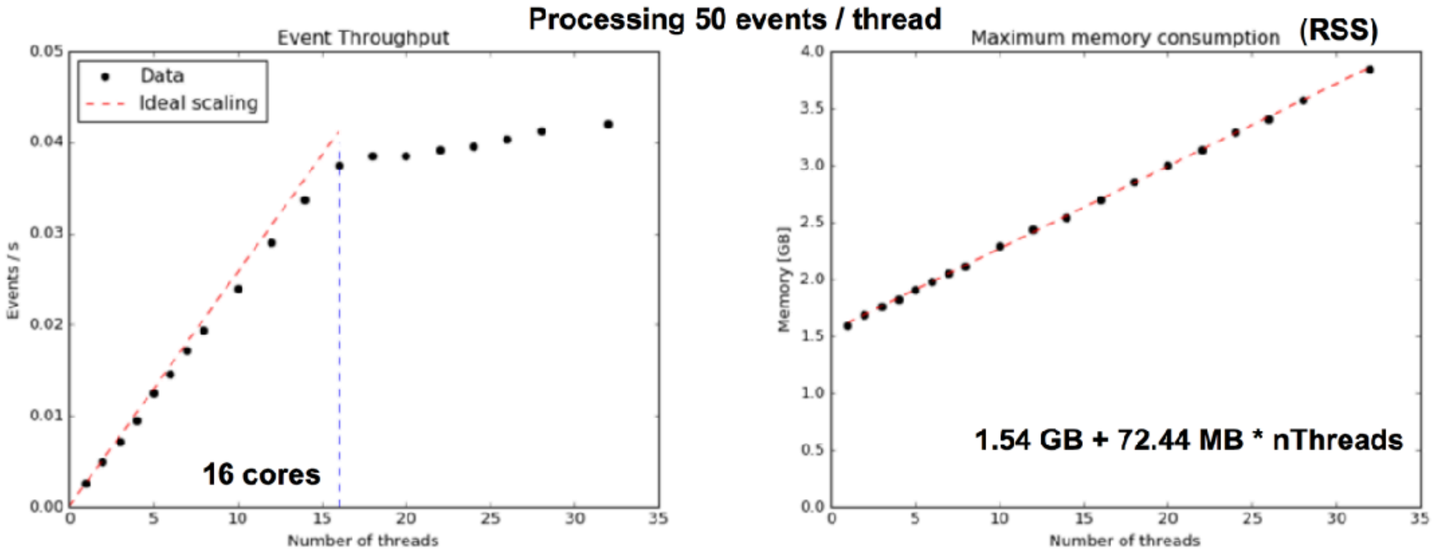
- Even this simpler case is not so simple!

- Core infrastructure (run managers, algos) => ***done***
- Thread initialization/finalization => ***done***
- Geometry, physics => ***done***
- Sensitive detectors => ***done except for LAr***
- User actions => ***done***
  - May see simplification in the future
- Magnetic field => ***done***
  - except for special very-forward case
- Writing truth => ***done***
  - Will see further development, though
- Frozen showers => ***done***
- FastCaloSim, FATRAS => ***not started***
- ISF-MT => ***in development***

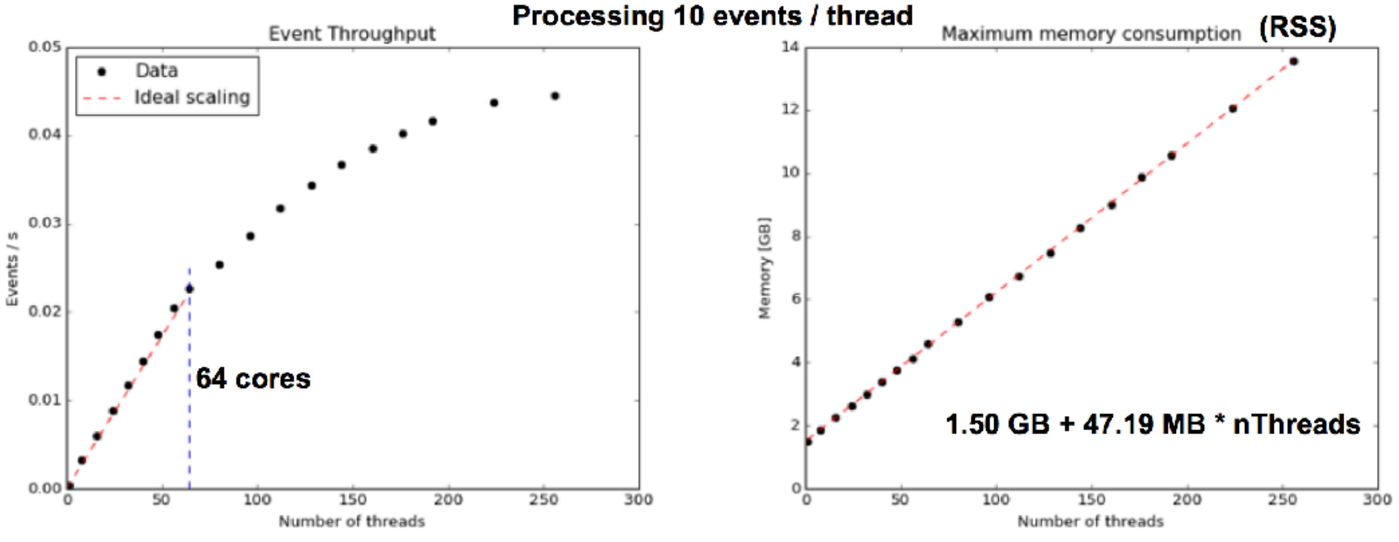
Scaling So Far

- Memory scaling is really excellent
- First really serious ATLAS code to run on a Xeon Phi
  - Some bottlenecks in faster event simulation — StreamHITS is serial
  - Much higher cycles per instruction than normal server (3.0 vs. 0.9)
    - Under investigation

Standard Xeon Zττ



Xeon Phi Zττ



Platforms

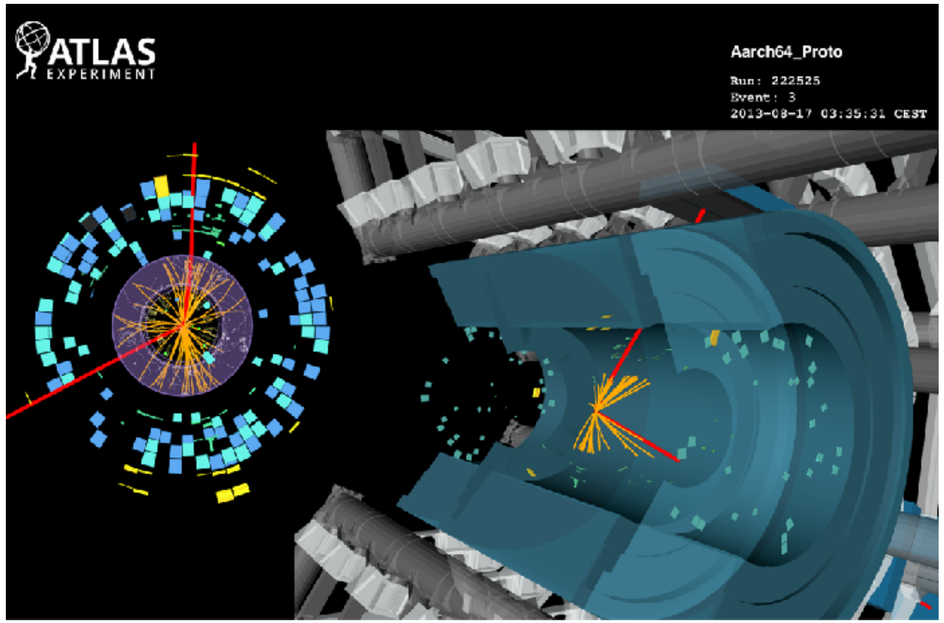
- SLC6 is looking rather old now
- CentOS7 is coming and we already have made progress here
  - SLC6 releases are validated for use on CentOS7 machines
  - Native builds on CentOS7 have started, but have not yet been a priority for developers (Release 21, git migration, …)
    - We do not anticipate many problems though
- We should have native builds for SLC6 and CentOS7 for Release 21

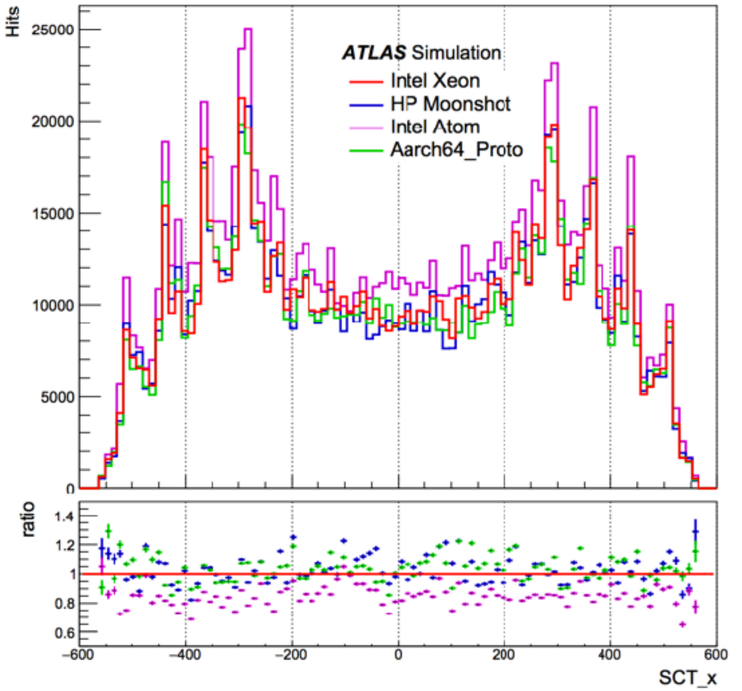
Mainly issues with TDAQ version

rel_	x86_64-centos7-gcc62-opt	13	22.0.0	100	35 (398)	01/19 04:46	97.5	70.7 (66.0)	N/A	01/19 05:03	01/19 04:54	01/19 04:54
rel_4	x86_64-slc6-gcc49-dbg	13	22.0.0	100	7 (329)	01/19 04:29	80.2	90.7 (84.8)	N/A	01/19 04:31	N/A	N/A
rel_4	x86_64-slc6-gcc49-opt	13	22.0.0	100	6 (328)	01/19 03:34	90.0	86.1 (79.9)	N/A	01/19 04:27	01/19 03:46	01/19 03:46

Platform Diversity

- Monoculture isn't very healthy for the code or the infrastructure
- Active efforts have begun to port Atlas Simulation release to Aarch64
- Port was achieved last year and results were very encouraging
  - Simulated ttbar events on ARM 64 and compares well with x86_64 results

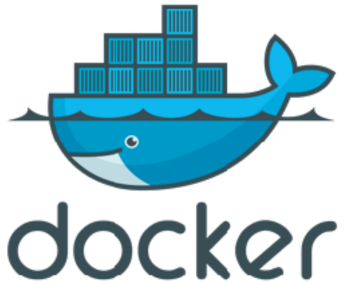




Future of Ports?

- Porting code turned out to be relatively easy
  - Helped greatly by the move to a modernised infrastructure
- We will be working on Aarch64 and adding this into the regular build system
- OpenPOWER is probably the next port we will tackle
  - Target the next generation of supercomputers with PowerPC chips (at least running in little endian mode — big endian is an issue for ROOT)
- However, proper validation of builds on different architectures is a time consuming business
  - We need to see how to do this, but it's inevitable there is going to be a threshold for use in production

Releases and Containers



- CVMFS had revolutionised software distribution
  - As mentioned before we'll rely on it more for testing nightly builds
- However, it doesn't satisfy all use cases
  - Developer environment is still very tied to the platforms we run on
    - SLC6 is not a very nice environment in which to develop
  - A few sites may not have CVMFS
    - Particularly HPC sites
- We are investigating how Docker can help us overcome this
  - Still in early stages, but encouraging
    - For developers hampered by the poor performance of volume mounts (like /cvmfs)
    - A possible solution here is a fat container with a whole release built into it
      - Technical investigation underway

Summary and Outlook

- ATLAS Software has a large program of work underway
  - Infrastructure modernisation is underway with the most radical change for developers about to happen
  - Much superior development workflows are available
    - A lot of increased flexibility for development, deployment, platforms
  - This retooling will help us hugely with the migration to a multi-threaded version of Athena
    - Simulation already making excellent progress
    - Workshop in March: https://indico.cern.ch/event/573143/
- In addition there will always be new ideas to improve physics performance
  - Especially in the light of increasing LHC luminosity