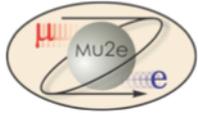


Mu2e-doc-1127-v3



art: A Framework For New, Small Experiments at Fermilab

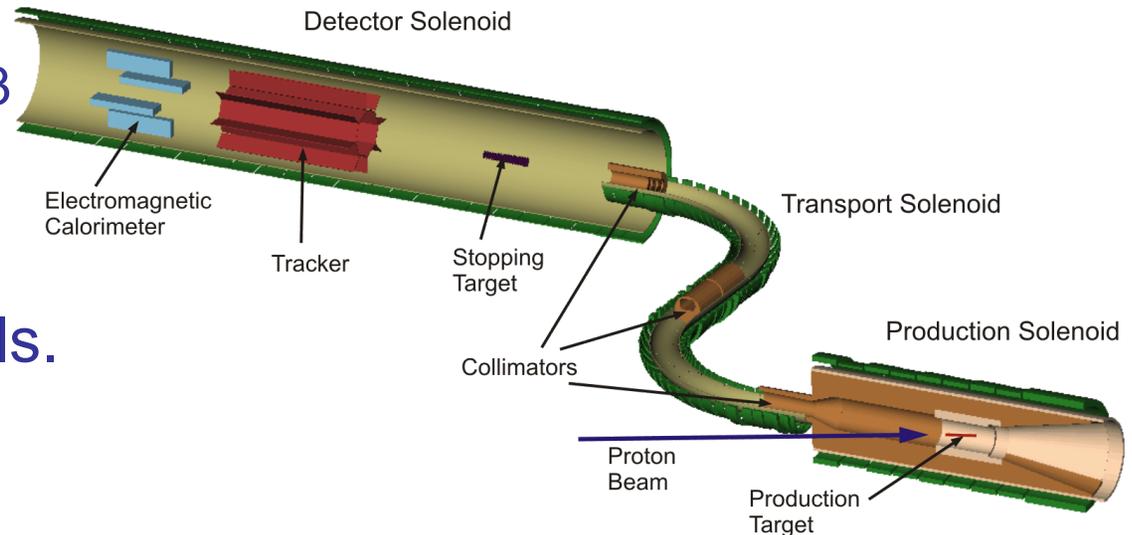
Rob Kutschke,
Fermilab CD and the [Mu2e](#) Collaboration
CHEP 2010, Academia Sinica, Taipei
October 21, 2010

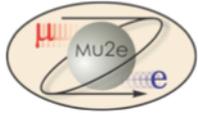


Mu2e



- A proposed experiment, at Fermilab, to search for the coherent, neutrino-less conversion of a muon into an electron in the Coulomb field of a nucleus.
 - $10^4 \times$ better sensitivity than SINDRUM II
 - Sensitive to mass scales of 10^4 TeV.
- Working schedule:
 - Construction start 2013
 - First data 2018.
- mu2e.fnal.gov
- $\approx 25k$ readout channels.



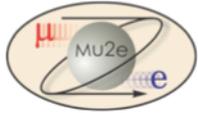


Infrastructure Software



- **Framework**
 - The state machine that drives the event loop.
 - Services that are integral to the framework proper.
 - EDM in memory
 - Run-time configuration.
- EDM persistency
- Build management
- Release management
- Workflow management, including GRID
- File catalog
- Databases
- Does not include:
 - Geometry, Conditions, Event Data classes, G4, reconstruction algorithms ...

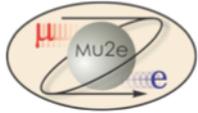
Framework is the glue that keeps all this together.



The Beginnings



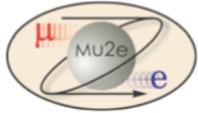
- Fall 2008: Mu2e needed infrastructure software:
 - Supported by FNAL Computing Division (CD).
 - Use cases: analysis, reconstruction, calibration, simulation, lowest non-real-time level of DAQ/Monitoring.
 - Physicists see analysis first; has to be teachable.
- Similar requests from
 - MicroBoone, NOvA, muon(g-2)
- CD willing to provide this but ...
 - O(2 FTE) for development and support.
 - Actually O(1 FTE) until mid 2010.
 - **Principals: Jim Kowalkowski, Marc Paterno.**



The Candidates



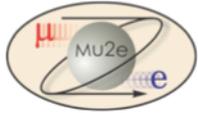
- Existing FNAL CD supported products:
 - D0, CDF, CMS, MiniBoone, MINOS ...
- Third party products
 - FMWK (ROOT based; MIPP, early NOvA)
 - ALIROOT / ILCROOT family
 - GAUDI
 - JAS
- With O(2 FTE), CD cannot support a third party product.
- **CD recommended evolving the CMS framework**
 - Most modern of the 5 FNAL candidates.
 - C++ based.
 - Remove/replace features if of little benefit to small experiments and hard to use and/or maintain.



Timeline



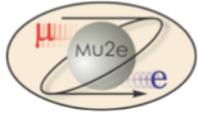
- Jan 2009:
 - Forked from CMSSW: framework + persistency.
 - Extract needed parts; dummy out some others: 4 people 1 week.
 - scones for build management
 - CMS wiki for documentation (but now private!)
- Used by Mu2e immediately
 - Integrated with G4.
 - Documentation:
<http://mu2e.fnal.gov/public/hep/computing/gettingstarted.shtml>
- Since May 2010
 - MicroBoone completed switchover a few weeks ago.
 - NOvA port well underway.
 - 1 person each with a low duty cycle.



Retained Features



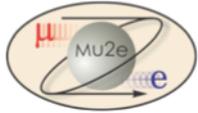
- The state machine.
- Module (Producer/Analyzer/Filter/IO) and Service base classes.
- Three part event ID
- EDM: in memory and ROOT Tree based persistency.
- Persistent objects: Event/Run/SubRun (SubRun=LuminosityBlock).
- Four part data product ID.
- ParameterSet mechanism.
- Python runtime config – will change.
- Reconstruction on demand.
 - Scheduled reconstruction also retained for now.
- Data product provenance.
- Exception handling strategy: action orthogonal to throw
- TFileService
- Message logger



Features (to be) Removed



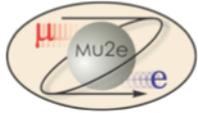
- EventSetup
 - Dummied out and will be removed.
 - Conditions data can be adequately managed via Services.
- Event merging/overlay
 - Removed from Source modules; now done in producers.
 - Factor into a bookkeeping problem and a physics problem.
- References across data products
 - Will develop something similar to CLEO III Lattice.
 - This puts the complexity in the right place.
- Matching module names to .so file name
 - CMS build system maintains a database.
 - We have a much smaller problem that will admit a simpler solution



Rolled Our Own



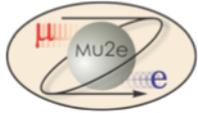
- Running G4 with framework driving the event loop.
- RandomNumberGeneratorService.
 - Permit more than one Engine per module.
 - More formal interaction with Module c'tor to reduce the possibility of an ill-defined state.
- Documentation



Standards and Practices



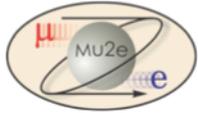
- Data products must not contain pointers.
 - Plan to develop CLEO III-like Lattice.
 - They contain (data product ID, index).
- Encourage ParameterSet default values in code.
- Very few reasons to use bare pointers (ROOT, G4).
- Event generators are Producers not Sources.
- Ask for data products by module label of creator
 - Enables reconstruction on demand.
- If unsure, throw.
 - Don't print a warning and carry on.
 - Important for a rapidly evolving detector and algorithms.
- Use `at(i)` not `[i]` for `std::vector` random access.



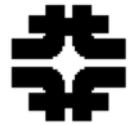
Refactoring and Development



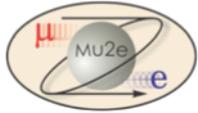
- Refactoring
 - Breaks backwards compatibility with CMSSW.
 - Since \approx Sept 1, 2010.
 - \approx 4 people, dedicated 2 or 3 days every 2 weeks.
 - Goal Dec 1, 2010
 - Break unnecessary couplings and remove obsolete code
 - Compatibility with old CMS data formats.
 - MessageLogger can be built separately.
 - Remove EventSetup.
- New Features
 - Reconfigure and Replay
 - Polymorphic views of data products
 - Change build system to cmake
 - New run-time configuration language



Multi-Threading



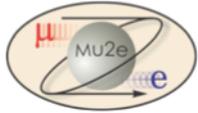
- Critical to make best use of future machines.
- Framework itself is thread-safe.
 - Ready to do module parallel execution.
 - But ROOT and G4 are not thread-safe (yet – but maybe soon?).
- Where might we use module parallel execution?
 - Non-real-time parts of the DAQ/monitoring world.
 - After G4, many types of hits need to be turned into digis.
 - All analysis modules may be executed in parallel.
- Future research direction:
 - Use framework as test bed to study sub-event parallelism.



Features I Like



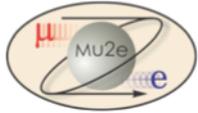
- Strong const and type safety.
- Strong audit trail.
- Reconstruction on demand.
- EDM: transient and persistent orthogonal.
- Multiple instances of one module in one job
 - Instances have different run time configuration.
- TFileService: Histograms in directories per module instance.
- Exceptions: throw and action are orthogonal.
- Information from Event, Services and Framework via handles:
 - Physicists do not check return codes but handles can throw.
- Multiple output files with runtime configurable content.
- FileInPath
- Most things just worked.
- But ... I would have liked much better documentation.



New User Experience - 1



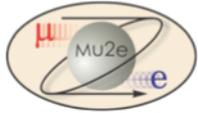
- Those who have some experience with modern frameworks and ROOT:
 - Few new ideas, just new syntax.
 - Most are productive in hours to 1 day.
- “Old Professor”:
 - Knows what he wants to investigate but ...
 - Does not know C++ or ROOT (or even C).
 - No one has put in more than ≈ 15 hours; not enough.
- “New student” working for “Old Professor”
 - Mixed results; depends strongly on the student.
 - OK if they “get” scientific computing.
 - We need much better introductions to C++ and unix.
 - Any suggestions?



New User Experience - 2



- Why do people have difficulty?
- Too many new ideas at once:
 - C++
 - STL
 - ROOT
 - CLHEP
 - G4
 - Framework
 - EDM
 - Choices Mu2e made about using the above.
 - Mu2e code



New User Experience - 3



- Access to higher level objects: use handles.
- Lower level interactions with objects use references:

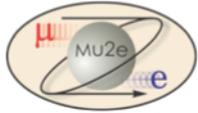
```
// Getting this information uses handles.
```

```
TTracker const& tracker = ....; // The geometry of the tracker  
StrawHit const& hit     = ....; // A simulated hit in the tracker.
```

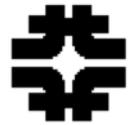
```
// Get the straw information:
```

```
Straw const &          straw = tracker.getStraw( hit.strawIndex() );  
CLHEP::Hep3Vector const & mid  = straw.getMidPoint();  
CLHEP::Hep3Vector const & w    = straw.getDirection();
```

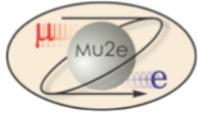
- The **&** is invisible even to experienced users
 - Unnecessary copies all over their code.



Odds and Ends



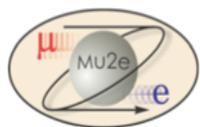
- Works on SLF4 and SLF5
 - Expect to port to Mac OS
 - No plans for a Windows port.
- What does the name “art” stand for?
 - Nothing
 - Originally “A Reconstruction Toolkit” but our vision is much broader than reconstruction and I think that a reconstruction toolkit has Kalman filters and cluster finders not just the bookkeeping tools.



Summary and Conclusions

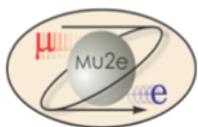


- art forked from CMSSW
- First release January 2009
 - Used by Mu2e since then.
 - Now used by MicroBoone
 - NOvA likely to adopt it soon.
- Commitment from CD to support O(2 FTE).
 - Realized for the past few months.
- Major refactoring in progress.
- New features to be added post refactoring.
- Will use art to study multi-threading.

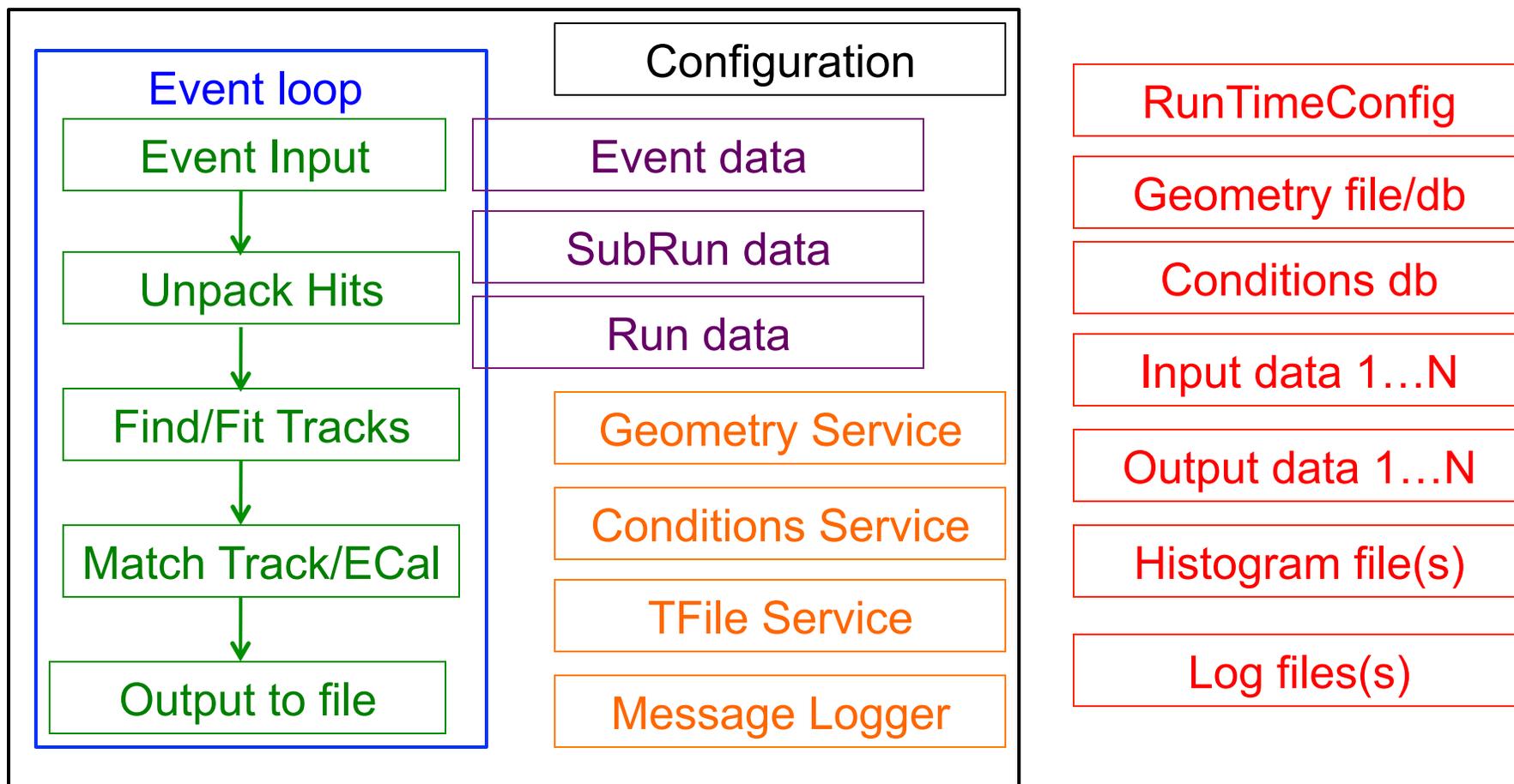


Backup and Working Slides

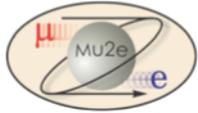




Major Elements



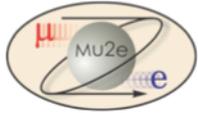
Framework **Modules** **Services** **Files/DB** Data in Memory



Events, Modules, Services



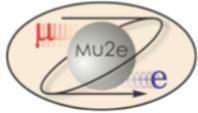
- Three part event ID
 - Run/SubRun/Event
 - Event holds “Data Products”.
- Module
 - Per event methods: analyze/produce/filter
 - begin/end: Job/Run/SubRun. Open/close: File
 - Communicate with other modules only via the event.
- Service
 - Singleton-like: lifetime and configuration managed by framework.
 - Some user provided: Conditions, Geometry,
 - Some provided by art: tracer, timer, memory use profile,



Not Yet Addressed



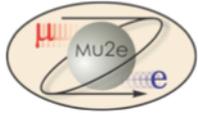
- Users will notice these two things but we have not had the time to think about them.
- ROOT IO speed
 - Plan to build on experiences of others.
 - Develop standards and practices for event-data objects.
- Data size
 - More of an experiment specific problem.
 - Develop advice for the experiments.



Status of Mu2e Software



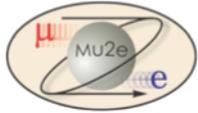
- Beamline and detector implemented.
 - Missing Cosmic Ray Veto sensitive volumes.
 - All else present, but missing some details.
 - 3 different Trackers: T(default), L, I.
 - 3 variants on T tracker.
- Hits made in calorimeter and straws.
 - Without detailed simulation of the electronics.
- Track finding underway.
- Clustering in calorimeter underway.
- G4 Graphics.
- Planning for CD1 review in March 2011.
 - CD1 \approx Conceptual Design



Why Drop Python?



- CD group would like to support a common configuration language for several projects, including this framework.
 - Other projects have rejected python.
- Python file is not a run-time configuration. It is a program to compute a run-time configuration. Actual run-time configuration may depend on the environment
- Would like the configuration file to be the actual config file, not the source code for something that computes the configuration.
 - Fits better with their view of the audit trail.



Wish List



- Compile time switch to enable run-time bounds checking in STL containers.
- An introduction to C++
 - Examples relevant to scientific programming
 - Teaches best practices
- Rudiments of unix
 - What is a shell
 - Environment
 - `.profile/.login` vs `.tcshrc/.bashrc`