

Mu2e-doc-1127-v1



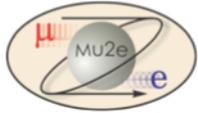
ART: A Framework For New, Small Experiments at Fermilab

Rob Kutschke,

Fermilab CD and the [Mu2e](#) Collaboration

CHEP 2010, Academia Sinica, Taipei

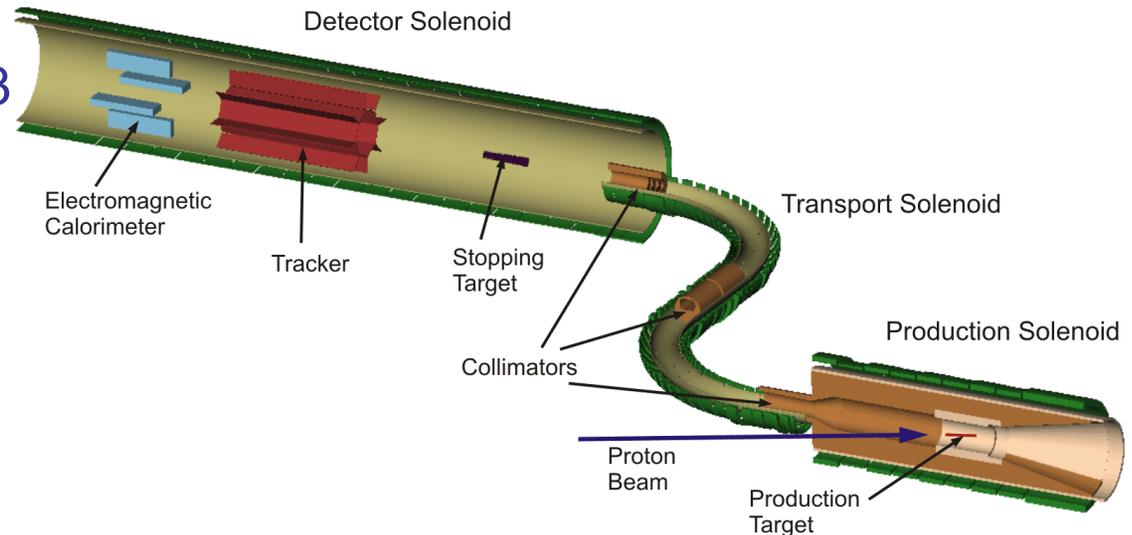
October 21, 2010

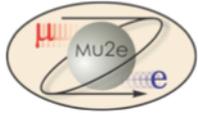


Mu2e



- 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



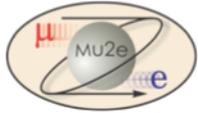


Infrastructure Software



- **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 ...

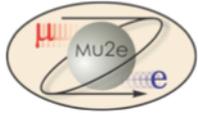
Framework is the glue that keeps all this together.



The Beginnings



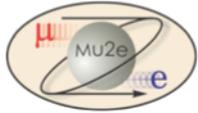
- Fall 2008: Mu2e needed infrastructure software:
 - Supported by FNAL Computing Division (CD).
 - Lowest non-real-time level of DAQ/Monitoring, simulation, reconstruction, analysis.
- Similar requests from
 - MicroBoone, NOvA, muon(g-2)
- CD willing to provide this but ...
 - O(2 FTE) for development and support.
 - Actually O(1 FTE) until mid 2010.
 - Leverage experience: D0, CDF, CMS, MiniBoone, MINOS
 - **Principals: Jim Kowalkowski, Marc Paterno.**



The Candidates



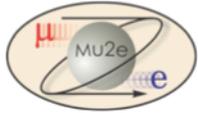
- Existing CD supported products:
 - D0, CDF, CMS, MiniBoone, MINOS ...
- Third party products
 - FMWK (ROOT based; MIPP, early NOvA)
 - ALIROOT / ILCROOT family
 - GAUDI
 - JAS
- With O(2 FTE), CD cannot support a third party product.
- **CD recommended evolving the CMS framework**
 - Most robust and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



Timeline



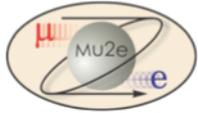
- Jan 2009:
 - Forked from CMS: framework + persistency.
 - Extract needed parts; dummy out some others: 4 people 1 week.
 - scon for build management
 - CMS wiki for documentation (but now private!)
- Used by Mu2e immediately.
 - Integrated with G4.
 - Documentation:
<http://mu2e.fnal.gov/public/hep/computing/gettingstarted.shtml>
- Since spring 2010
 - MicroBoone has ported their code.
 - NOvA port well underway



Retained Features



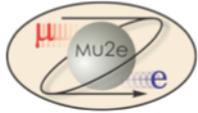
- The state machine.
- Module (Producer/Analyzer/Filter/IO) and Service base classes.
- Three part event ID
- EDM: in memory and ROOT persistency.
- Four part data product ID.
- ParameterSet mechanism.
- Python runtime config – will change.
- Reconstruction on demand.
 - Scheduled reconstruction retained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



Features (to be) Removed



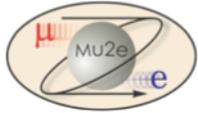
- EventSetup
 - Dummied out and will be removed.
 - Conditions can be adequately managed via Service call back mechanism.
- Event merging/overlay
 - Removed from Source modules; done in producers.
 - Factor into a bookkeeping problem and a physics problem.
- References across data products
 - Will develop something similar to CLEO III Lattice.
 - This puts the complexity where it belongs.
- Module registration database
 - We believe we have a much simpler problem that will admit a simpler solution.
 - (What is the proper name for this).
- What else ... ?



Rolled Our Own



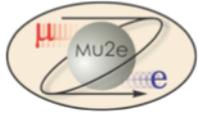
- Stealing the event loop from G4.
- RandomNumberGeneratorService.
 - Permit more than one Engine per module.
 - More formal interaction with Module c'tor to reduce the possibility of an ill-defined state.
- Documentation.



Standards and Practices



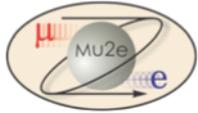
- Encourage ParameterSet defaults in code.
- Event generators are Producers not Sources.
- Data products must not contain pointers.
 - They contain (data product ID, index)
 - Plan to develop CLEO III-like Lattice.
- Do your job right or throw.



Refactoring and Development



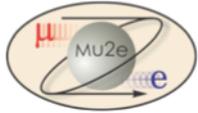
- Refactoring:
 - 4 people, dedicated 2 or 3 days every 2 weeks.
 - Goal Dec 1, 2010
 - MessageLogger now a separate product
 - Remove EventSetup.
- New Features
 - Reconfigure and Replay
 - New configuration language.



Refactoring and Development



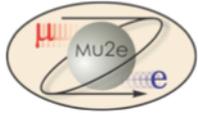
- Refactoring:
 - 4 people, dedicated 2 or 3 days every 2 weeks.
 - Goal Dec 1, 2010
 - MessageLogger now a separate product
 - Remove EventSetup.
- New Features
 - Reconfigure and Replay
 - Polymorphic views of data products.
 - Changing build system to cmake.



Not yet Addressed



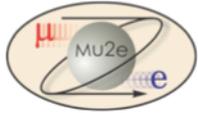
- ROOT IO speed
 - Plan to build on experiences of others.
 - Develop standards and practices.
- Data size
 - More of an experiment specific problem.



Multi-Threading



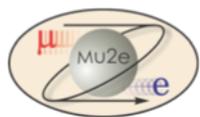
- Framework itself is thread safe.
 - But ROOT and G4 are not.
- A candidate for use in non-real-time parts of the DAQ/monitoring world.
- Reconstruction



Features



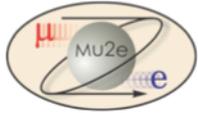
- Strong const and type safety
- Strong audit trail.
 - Includes run time configuration.
- Orthogonality of exception throw and response.
- Reconstruction on demand or scheduled reconstruction.



These Comments need a home



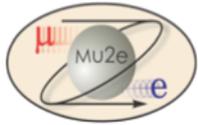
- Analysis first; work backwards.
- Compromise
 - What to push onto physicists (since finite support)
- Audit trail automated
- Exception safe
- Use as a platform to study multi-threading.
 - Potential online applications.
- Works on Linux, expect to port to Mac; no plans for a windows port.
- What does the name mean”
 - A Reconstruction Toolkit
 - but “Reconstruction” too limiting so it does not really mean anything.



Summary and Conclusions

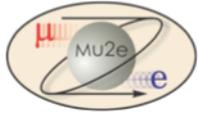


- ART derived from a snapshot of CMSsoft
- First release January 2009
 - Used by Mu2e since then.
 - Now used by MicroBoone
 - NOvA likely to adopt it soon.
- Commitment from CD to support O(2 FTE).
- Major refactoring in progress.
- New features to be added post refactoring.
- Will use ART to study multi-threading.



Backup and Working Slides

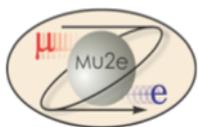




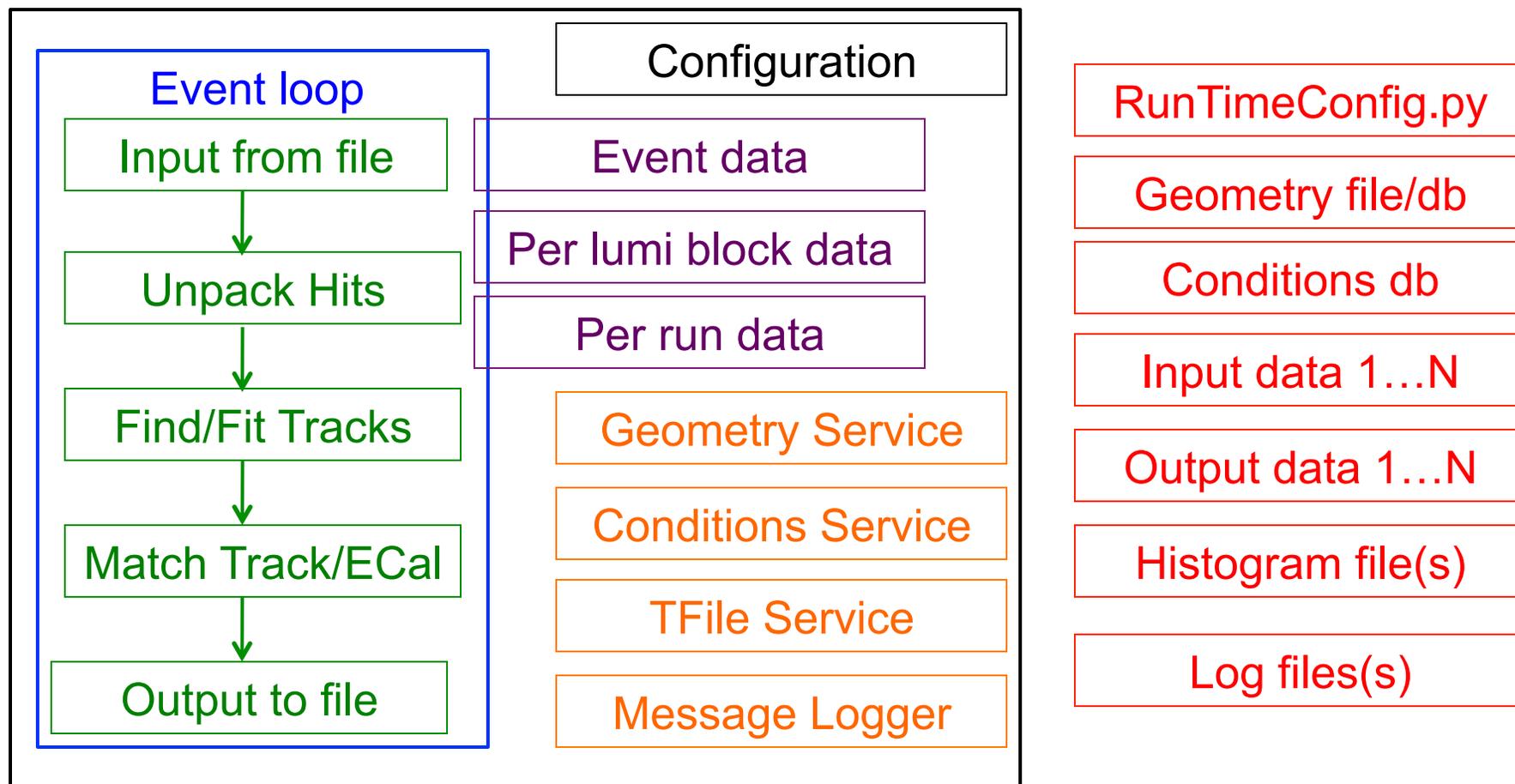
Why Drop Python?



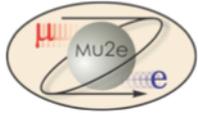
- CD group would like to support a common configuration language for several projects, including this framework.
 - Other projects have rejected python.
- Python file is not a run-time configuration. It is a program to compute a run-time configuration. Actual run-time configuration may depend on the 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.



Major Elements of the Framework



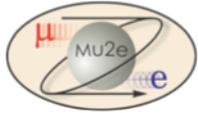
Framework **Modules** **Services** **Files/DB** **Data in Memory**



Events, Modules, Services



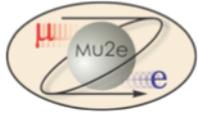
- Three part event ID
 - Run/SubRun/Event
 - Event holds “Data Products”.
- Module
 - Per event methods: analyze/produce/filter
 - begin/end: Job/Run/SubRun. Open/close: File
 - Communicate with other modules only via the event.
- Service
 - Singleton-like: lifetime and configuration managed by framework.
 - Some user provided; some provided by ART.



Why do I like it



- Reconstruction on demand
- Tools in place for a strong audit trail
- EDM: Transient/Persistency orthogonal
- Throw/Action orthogonal.
- Exception safe
- User interaction with framework and EDM via handles; throw if you do something wrong. Physicists do not need to check return codes.
- Many things “just worked”
- Would have liked better documentation.



Do I want to make These Points?



- Lots of failed attempts to build common frameworks. Other than some derivative works this has failed. Why will this one work?
- Design run time config so that if I run N jobs, there is a one authoritative run time config, with a unique checksum, and N additional config fragments that inject the information that is per-job specific. Presumably WMS injects these fragments. Want WMS to know as little as possible.