The art Framework

Chris Green
Fermilab Scientific Software
Infrastructure Group
CHEP 2012
21 May, 2012

Fermi National Accelerator Laboratory
Office of Science / U.S. Department of Energy
Managed by Fermi Research Alliance, LLC

Outline

- What is **art**? Why is **art**?
- Architecture & key features.
- Origins of **art**.
- Collaborative development.
- The Future.
- Summary.

- What is art?

- What is **art**?
  art is a generic C++-based modular analysis framework, for use from generator-level or DAQ event building through simulation, production and user analysis. **art** grew out of the CMS framework and was developed to satisfy the common requirements of intensity frontier experiments (initially **Mu2e**, **NOνA** and **LArSoft**).

- What is **art**?

  art is a generic C++-based modular analysis framework, for use from generator-level or DAQ event building through simulation, production and user analysis. **art** grew out of the CMS framework and was developed to satisfy the common requirements of intensity frontier experiments (initially **Mu2e**, **NOνA** and **LArSoft**).
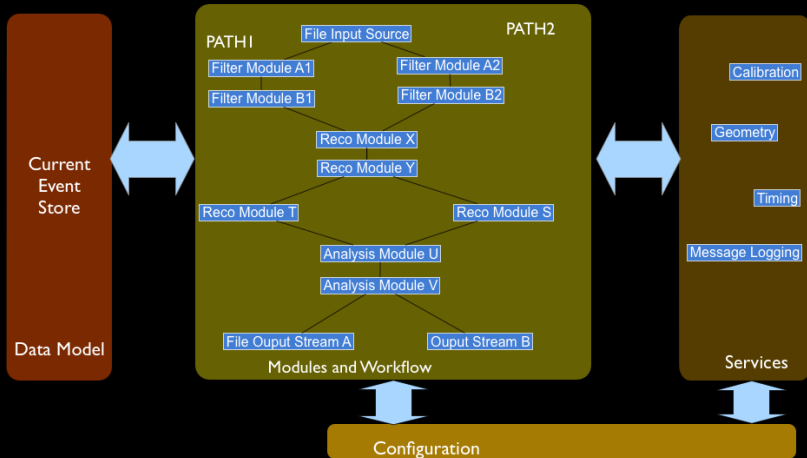
- Why is **art**?

- What is **art**?
  art is a generic C++-based modular analysis framework, for use from generator-level or DAQ event building through simulation, production and user analysis. **art** grew out of the CMS framework and was developed to satisfy the common requirements of intensity frontier experiments (initially **Mu2e**, **NOνA** and **LArSoft**).

- Why is **art**?
  Most HEP experiments use a framework; **art** is a framework that is being used by multiple experiments, which has relieved them of the need to produce and maintain their own.

HEP Framework

PATH1   File Input Source   PATH2

Filter Module A1   Filter Module A2

Filter Module B1   Filter Module B2

Reco Module X

Reco Module Y

Reco Module T   Reco Module S

Analysis Module U

Analysis Module V

File Ouput Stream A   Ouput Stream B

Modules and Workflow

Current Event Store

Data Model

Calibration

Geometry

Timing

Message Logging

Services

Configuration

```
#snip
source: {
  module_type: RootInput
  fileNames: [ "file1.root",
               "file2.root" ]
}
physics.producers.trac1: {
  module_type: TrackFinder
  myPar: 5
}
physics.producers.trac2: {
  module_type: TrackFinder
  myPar: 10
}
#snip
```

Architecture

- Experiments use **art** as an external package – their build system is not tied to that used to develop **art**.

Architecture

- Experiments use **art** as an external package – their build system is not tied to that used to develop **art**.
- I/O and work schedule are handled by a state machine.

Architecture

- Experiments use **art** as an external package – their build system is not tied to that used to develop **art**.
- I/O and work schedule are handled by a state machine.
- Modules are generally provided by users, and are divided into inputs (**sources**), **producers**, **filters**, **analyzers** and **outputs**.

Architecture

- Experiments use **art** as an external package – their build system is not tied to that used to develop **art**.
- I/O and work schedule are handled by a state machine.
- Modules are generally provided by users, and are divided into inputs (**sources**), **producers**, **filters**, **analyzers** and **outputs**.
- Inter-module communication is handled principally by means of persistent data structures (**products**) passed via entities with known lifetimes: **event**, **subrun**, **run**.

Architecture

- Experiments use **art** as an external package – their build system is not tied to that used to develop **art**.
- I/O and work schedule are handled by a state machine.
- Modules are generally provided by users, and are divided into inputs (**sources**), **producers**, **filters**, **analyzers** and **outputs**.
- Inter-module communication is handled principally by means of persistent data structures (**products**) passed via entities with known lifetimes: **event**, **subrun**, **run**.
- **products** are distinguished from algorithms $\implies$ modules don't need to address persistency mechanics.

Architecture

- Experiments use **art** as an external package – their build system is not tied to that used to develop **art**.
- I/O and work schedule are handled by a state machine.
- Modules are generally provided by users, and are divided into inputs (**sources**), **producers**, **filters**, **analyzers** and **outputs**.
- Inter-module communication is handled principally by means of persistent data structures (**products**) passed via entities with known lifetimes: **event**, **subrun**, **run**.
- **products** are distinguished from algorithms $\implies$ modules don't need to address persistency mechanics.
- **products** retrieved from the data store are non-modifiable: derived or edited data are saved as a new product.

Architecture

- Experiments use **art** as an external package – their build system is not tied to that used to develop **art**.
- I/O and work schedule are handled by a state machine.
- Modules are generally provided by users, and are divided into inputs (**sources**), **producers**, **filters**, **analyzers** and **outputs**.
- Inter-module communication is handled principally by means of persistent data structures (**products**) passed via entities with known lifetimes: **event**, **subrun**, **run**.
- **products** are distinguished from algorithms $\implies$ modules don't need to address persistency mechanics.
- **products** retrieved from the data store are non-modifiable: derived or edited data are saved as a new product.
- Configurable exception handling: categorization of a failure is distinct from its handling action.

- Facility for products to refer to other products in collections already saved (`Ptr`).

Key features

- Facility for products to refer to other products in collections already saved (**Ptr**).
- **product** mixing ("pile-up"): users need to know how to combine the data from multiple instances of a particular **product**, but not the mechanics of obtaining those data and writing out the merged **product**.

Key features

- Facility for products to refer to other products in collections already saved (**Ptr**).

- **product** mixing ("pile-up"): users need to know how to combine the data from multiple instances of a particular **product**, but not the mechanics of obtaining those data and writing out the merged **product**.

- Metadata may be stored in a relational **SQLite** database in memory and / or embedded in a **ROOT** data file.

Key features

- Facility for products to refer to other products in collections already saved (**Ptr**).
- **product** mixing ("pile-up"): users need to know how to combine the data from multiple instances of a particular **product**, but not the mechanics of obtaining those data and writing out the merged **product**.
- Metadata may be stored in a relational **SQLite** database in memory and / or embedded in a **ROOT** data file.
- Simple configuration language with partitioned module configuration information.

Key features

- Facility for products to refer to other products in collections already saved (**Ptr**).
- **product** mixing ("pile-up"): users need to know how to combine the data from multiple instances of a particular **product**, but not the mechanics of obtaining those data and writing out the merged **product**.
- Metadata may be stored in a relational **SQLite** database in memory and / or embedded in a **ROOT** data file.
- Simple configuration language with partitioned module configuration information.
- Bi-directional associations (**Assns**) between **products** already in the data store.

Key features

- Facility for products to refer to other products in collections already saved (`Ptr`).
- `product` mixing ("pile-up"): users need to know how to combine the data from multiple instances of a particular `product`, but not the mechanics of obtaining those data and writing out the merged `product`.
- Metadata may be stored in a relational **SQLite** database in memory and / or embedded in a **ROOT** data file.
- Simple configuration language with partitioned module configuration information.
- Bi-directional associations (`Assns`) between `products` already in the data store.
- An input source class template for more straightforward user implementation of "raw" data input.

Origins of art

Over the last 15 years, the **art** authors have been involved in writing multiple frameworks for HEP experiments: **DØ**, **BTeV**, **MiniBooNE**, **CMS**. **art** grew out of the **CMS** framework (forked in 2010).

Origins of art

Over the last 15 years, the **art** authors have been involved in writing multiple frameworks for HEP experiments: **DØ**, **BTeV**, **MiniBooNE**, **CMS**. **art** grew out of the **CMS** framework (forked in 2010).

Simplifications and tradeoffs:

- Simpler data products (storage of only concrete types).

Origins of art

Over the last 15 years, the **art** authors have been involved in writing multiple frameworks for HEP experiments: **DØ**, **BTeV**, **MiniBooNE**, **CMS**. **art** grew out of the **CMS** framework (forked in 2010).

Simplifications and tradeoffs:

- Simpler data products (storage of only concrete types).
- Removal of `EventSetup`.

Origins of art

Over the last 15 years, the **art** authors have been involved in writing multiple frameworks for HEP experiments: **DØ**, **BTeV**, **MiniBooNE**, **CMS**. **art** grew out of the **CMS** framework (forked in 2010).

Simplifications and tradeoffs:

- Simpler data products (storage of only concrete types).
- Removal of `EventSetup`.
- Simplification of build system (moved to **CMake**).

*Origins of *art**

Over the last 15 years, the **art** authors have been involved in writing multiple frameworks for HEP experiments: **DØ**, **BTeV**, **MiniBooNE**, **CMS**. **art** grew out of the **CMS** framework (forked in 2010).

Simplifications and tradeoffs:

- Simpler data products (storage of only concrete types).
- Removal of **EventSetup**.
- Simplification of build system (moved to **CMake**).
- Simplification of plugin system: rely on naming conventions(`_module.cc`, `_source.cc`, *etc.*) rather than build-generated runtime artifacts.

*Origins of **art***

Over the last 15 years, the **art** authors have been involved in writing multiple frameworks for HEP experiments: **DØ**, **BTeV**, **MiniBooNE**, **CMS**. **art** grew out of the **CMS** framework (forked in 2010).

Simplifications and tradeoffs:

- Simpler data products (storage of only concrete types).
- Removal of `EventSetup`.
- Simplification of build system (moved to **CMake**).
- Simplification of plugin system: rely on naming conventions(`_module.cc`, `_source.cc`, *etc.*) rather than build-generated runtime artifacts.
- New, simple configuration language, **FHiCL** to match stakeholder requirements replaces use of **Python** and associated **Python** modules. **FHiCL** is used by other projects such as **LQCD** and has **Python** and **Ruby** bindings.

- **art** is developed by a small team, with weekly input and priority setting from interested individuals on each experiment.

---

[1] https://redmine.fnal.gov/projects/art?jump=welcome
[2] art-users@fnal.gov, artists@fnal.gov

Collaborative development

- **art** is developed by a small team, with weekly input and priority setting from interested individuals on each experiment.
- Additional interaction via issue tracker on redmine[1], email lists[2].

---

[1] https://redmine.fnal.gov/projects/art?jump=welcome
[2] art-users@fnal.gov, artists@fnal.gov

Collaborative development

- **art** is developed by a small team, with weekly input and priority setting from interested individuals on each experiment.
- Additional interaction via issue tracker on redmine[1], email lists[2].
- Binary package delivery system:

---

[1] https://redmine.fnal.gov/projects/art?jump=welcome
[2] art-users@fnal.gov, artists@fnal.gov

Collaborative development

- **art** is developed by a small team, with weekly input and priority setting from interested individuals on each experiment.
- Additional interaction via issue tracker on redmine[1], email lists[2].
- Binary package delivery system:
  - Experiments are not constrained to use a particular build system to use **art**.

---

[1] https://redmine.fnal.gov/projects/art?jump=welcome
[2] art-users@fnal.gov,artists@fnal.gov

Collaborative development

- **art** is developed by a small team, with weekly input and priority setting from interested individuals on each experiment.
- Additional interaction via issue tracker on redmine[1], email lists[2].
- Binary package delivery system:
  - Experiments are not constrained to use a particular build system to use **art**.
  - **art** can be developed as multiple packages but treated as one due to automatic setup of dependencies.

---

[1] https://redmine.fnal.gov/projects/art?jump=welcome
[2] art-users@fnal.gov, artists@fnal.gov

Collaborative development

- **art** is developed by a small team, with weekly input and priority setting from interested individuals on each experiment.
- Additional interaction via issue tracker on redmine[1], email lists[2].
- Binary package delivery system:
    - Experiments are not constrained to use a particular build system to use **art**.
    - **art** can be developed as multiple packages but treated as one due to automatic setup of dependencies.
- Experiments develop their own modules, services, auxiliary code and (optionally) main programs which interact with **art**.

---

[1] https://redmine.fnal.gov/projects/art?jump=welcome
[2] art-users@fnal.gov, artists@fnal.gov

- Expand use of **SQLite** DB to all existing metadata.

- Expand use of **SQLite** DB to all existing metadata.
- Unify the concepts of **event**, **subrun** and **run**.

Future enhancements

- Expand use of **SQLite** DB to all existing metadata.
- Unify the concepts of **event**, **subrun** and **run**.
- Revamp processing intervals.

Future enhancements

- Expand use of **SQLite** DB to all existing metadata.
- Unify the concepts of **event**, **subrun** and **run**.
- Revamp processing intervals.
- Remove internal use of Reflex to be ready for **ROOT**/ **Cling**.

Future enhancements

- Expand use of **SQLite** DB to all existing metadata.
- Unify the concepts of `event`, `subrun` and `run`.
- Revamp processing intervals.
- Remove internal use of Reflex to be ready for **ROOT**/**Cling**.
- Move to **ISO C++ 2011** (already used in development, **artdaq**).

- Allow user-defined metadata in **SQLite** DB.

- Allow user-defined metadata in **SQLite** DB.
- Event display toolkit (graphical toolkit agnostic): better-defined / -suited interface to framework for operators, algorithm developers.

Coming attractions

- Allow user-defined metadata in **SQLite** DB.
- Event display toolkit (graphical toolkit agnostic): better-defined / -suited interface to framework for operators, algorithm developers.
- Generalize and expand **CMake**-based build / package delivery system for use by experiments as an alternative to supporting their own build system.

Future directions

- "Multi-schedule **art**": process multiple events simultaneously in the same executable; in addition, allowing for algorithm parallelization within modules.

---

[3]Message Passing Interface http://www.mcs.anl.gov/mpi/

Future directions

- "Multi-schedule **art**": process multiple events simultaneously in the same executable; in addition, allowing for algorithm parallelization within modules.
- Currently prototyping DAQ event-building and triggering using **art** (**artdaq**) in conjunction with **MPI**[3] for **DS50**, **Mu2e**, μ**BooNE**, **NOνA** experiments.

---

[3]Message Passing Interface http://www.mcs.anl.gov/mpi/

Future directions

- "Multi-schedule **art**": process multiple events simultaneously in the same executable; in addition, allowing for algorithm parallelization within modules.
- Currently prototyping DAQ event-building and triggering using **art** (**artdaq**) in conjunction with **MPI**[3] for **DS50**, **Mu2e**, μ**BooNE**, **NO**ν**A** experiments.
- Multi-thread and multi-process parallel I/O.

---

[3] Message Passing Interface http://www.mcs.anl.gov/mpi/

Summary

- **art** used currently by **g-2**, **LArSoft** (μ**BooNE**, **ArgoNeuT**, **LBNE**), **Mu2e**, **NO**ν**A** since early 2011. Enquiries from **SuperB**.

Summary

- **art** used currently by **g-2**, **LArSoft** (μ**BooNE**, **ArgoNeuT**, **LBNE**), **Mu2e**, **NOνA** since early 2011. Enquiries from **SuperB**.
- Supporting **art** mainstream development with <2 FTE.

- **art** used currently by **g-2**, **LArSoft** (μ**BooNE**, **ArgoNeuT**, **LBNE**), **Mu2e**, **NO**ν**A** since early 2011. Enquiries from **SuperB**.
- Supporting **art** mainstream development with <2 FTE.
- Early, encouraging results for **NO**ν**A** `DDT` using real cosmic data from near detector (see **NO**ν**A** `DAQ` poster).

Summary

- **art** used currently by **g-2**, **LArSoft** (μ**BooNE**, **ArgoNeuT**, **LBNE**), **Mu2e**, **NO**ν**A** since early 2011. Enquiries from **SuperB**.
- Supporting **art** mainstream development with <2 FTE.
- Early, encouraging results for **NO**ν**A DDT** using real cosmic data from near detector (see **NO**ν**A DAQ** poster).
- More information:

Summary

- **art** used currently by **g-2**, **LArSoft** (μ**BooNE**, **ArgoNeuT**, **LBNE**), **Mu2e**, **NOνA** since early 2011. Enquiries from **SuperB**.
- Supporting **art** mainstream development with <2 FTE.
- Early, encouraging results for **NOνA DDT** using real cosmic data from near detector (see **NOνA DAQ** poster).
- More information:
  - https://redmine.fnal.gov/projects/art?jump=welcome

Summary

- **art** used currently by **g-2**, **LArSoft** (μ**BooNE**, **ArgoNeuT**, **LBNE**), **Mu2e**, **NOνA** since early 2011. Enquiries from **SuperB**.
- Supporting **art** mainstream development with <2 FTE.
- Early, encouraging results for **NOνA** **DDT** using real cosmic data from near detector (see **NOνA** **DAQ** poster).
- More information:
  - https://redmine.fnal.gov/projects/art?jump=welcome
  - art-users@fnal.gov, community list.

- **art** used currently by **g-2**, **LArSoft** (μ**BooNE**, **ArgoNeuT**, **LBNE**), **Mu2e**, **NOνA** since early 2011. Enquiries from **SuperB**.
- Supporting **art** mainstream development with <2 FTE.
- Early, encouraging results for **NOνA** `DDT` using real cosmic data from near detector (see **NOνA** `DAQ` poster).
- More information:
  - https://redmine.fnal.gov/projects/art?jump=welcome
  - art-users@fnal.gov, community list.
  - artists@fnal.gov, expert advice list.

- **art** used currently by **g-2**, **LArSoft** (μ**BooNE**, **ArgoNeuT**, **LBNE**), **Mu2e**, **NOνA** since early 2011. Enquiries from **SuperB**.
- Supporting **art** mainstream development with <2 FTE.
- Early, encouraging results for **NOνA** **DDT** using real cosmic data from near detector (see **NOνA** **DAQ** poster).
- More information:
  - https://redmine.fnal.gov/projects/art?jump=welcome
  - art-users@fnal.gov, community list.
  - artists@fnal.gov, expert advice list.
  - http://mu2e.fnal.gov/public/hep/computing/gettingstarted.shtml

Summary

- **art** used currently by **g-2**, **LArSoft** (μ**BooNE**, **ArgoNeuT**, **LBNE**), **Mu2e**, **NOνA** since early 2011. Enquiries from **SuperB**.
- Supporting **art** mainstream development with <2 FTE.
- Early, encouraging results for **NOνA** `DDT` using real cosmic data from near detector (see **NOνA** `DAQ` poster).
- More information:
  - https://redmine.fnal.gov/projects/art?jump=welcome
  - art-users@fnal.gov, community list.
  - artists@fnal.gov, expert advice list.
  - http://mu2e.fnal.gov/public/hep/computing/gettingstarted.shtml
  - *NOνA Event Building, Buffering and Filtering From Within the DAQ system* poster at CHEP 2012.

Summary

- **art** used currently by **g-2**, **LArSoft** (μ**BooNE**, **ArgoNeuT**, **LBNE**), **Mu2e**, **NOνA** since early 2011. Enquiries from **SuperB**.
- Supporting **art** mainstream development with <2 FTE.
- Early, encouraging results for **NOνA DDT** using real cosmic data from near detector (see **NOνA DAQ** poster).
- More information:
  - https://redmine.fnal.gov/projects/art?jump=welcome
  - art-users@fnal.gov, community list.
  - artists@fnal.gov, expert advice list.
  - http://mu2e.fnal.gov/public/hep/computing/gettingstarted.shtml
  - *NOνA Event Building, Buffering and Filtering From Within the DAQ system* poster at CHEP 2012.
  - *Software for the **Mu2e** Experiment* poster at CHEP 2012.