Mu2e-doc-823-v1

Mu2e Uses of Run Time Configuration

Rob Kutschke, Fermilab

March 26, 2010

Mu2e on the Web

- Home page:
  - http://mu2e.fnal.gov/
- Software getting started:
  - http://mu2e.fnal.gov/public/hep/computing/gettingstarted.shtml
  - 5th bullet: “Running G4 ….”
- Code browser:
  - http://cdcvs0.fnal.gov/cgi-bin/public-cvs/cvsweb-public.cgi/mu2e/Offline/

Module and Service Config

```
process.checkhits = mu2e.EDAnalyzer(    "ReadBack",
                g4ModuleLabel = mu2e.string("g4run"),
                minimumEnergy = mu2e.double(0.001),
                maxFullPrint  = mu2e.untracked.int32(201)
)
```

```
process.GeometryService = mu2e.Service("GeometryService",
     inputfile=mu2e.untracked.string("Mu2eG4/test/geom_01.txt")
)
```

- We need a configuration language to configure modules and services.

Other Uses

- Need a config language for text files that are place holders data that will end up in data bases.
  - smallish ≈O(300) numbers.
- Distinct uses:
  - Nominal geometry – becoming complete.
  - Conditions data – very little now, lots and lots later.
- Manage provenance with cvs.
- We now use an evolution of Rob's first C++ project.
- Can this be the same language as the framework config?.

Geometry File

```
// LTracker: Positioning info is relative to DS.
int    ltracker.nSides              =     8;
double ltracker.r0                  =   388.48435399; // mm
double ltracker.z0                  = 10200.;         // mm
double ltracker.phi0                =     0.0;        // radians
double ltracker.tilt                =     0.015;      // radians
double ltracker.strawThickness      =     0.025;      // mm
double ltracker.rWire               =     0.010;      // mm
double ltracker.carbonThick         =     1.e-4;      // =100 nm
vector<int> ltracker.nStrawsSide    = { 63, 64, 65 };
vector<double> ltracker.vaneOffset = { 572., 10., 0.};  // mm
```

- Not real structures. Just suggestive names.
- (int, double, string, bool) and vector<> of the first three.
- No include capability
- If a key appears twice it is an error.
- It would be nice to convert to standard units on input.

API

```
// Key must be present in config
_nSides        = config.getInt("ltracker.nSides");

// If key not present, then default is used;
_r0            = config.getDouble("ltracker.r0”, 400.);
```

// Syntax for vector-valued objects. No syntax for a default
 vector<int> nStrawsSide;
config.getVectorInt("ltracker.nStrawsSide", nStrawsSide);

// This syntax requires that there be exactly three items in the vector.
config.getVectorInt("ltracker.nStrawsSide", nStrawsSide, 3);

Wish List

- Can both config uses use the same tool?
  - Nice but not required.
- A grammar to say:
  - Do the standard thing except for a these exceptions
  - Track this.
- Require units to be specified and convert to MeV/mm/ns/radians on input.
  - Error if units do not match.

Examples of Actual Config Files

- Offline/Mu2eG4/test/
  - g4test_03.py
  - readback.py
  - geom_01.txt
  - genconfig_01.txt
  - conditions_01.txt

Slides added Afterwards

- These shows a candidate syntax for a unit-full configuration system.

Geometry File

```
// A candidate for the syntax for unit-full numbers.
int    ltracker.nsides                = 8;
double ltracker.r0                    =   “388.48435399 mm”;
vector<double> ltracker.vaneOffset = “{ 572., 10., 0.} mm”;
vector<double> ltracker.vaneOffset =
                        “{ 1. mm^{-1}, 1. radian, 1. mm}”;
```

- nsides is dimensionless so units must not be present.
- Prefer syntax without quotes but OK if the solution would take too long.
- Fourth syntax is a very low priority.
  - The use is for things like state vectors for a track fitter.
  - Possibly we could use an non-unit-safe accessor for this?

API

```
// Suggestion for unit-safe get syntax?
double r0 = config.get<Length>("ltracker.r0”, 400.);
double minimumEnergy = config.get<Energy>( “minimumEnergy”);
```