PROGRESS WITH TRS
TpcResponseSimulator

Brian Lasiuk

May 28, 1998

- **STAR Class Library**
  - update/summary
- **Data Base Issues In TRS**
  - topical subject
- **Coordinate Transformations**
  - a general approach
- **Pad Response Generation**
  - preliminary calculations

STAR Class Library

- **global**
  - define types and units (CLHEP and extensions)
- **vectors+matrices**
  - templated vectors and matrices
  - interface compatible with CLHEP
- **random**
  - CLHEP generators (5)
  - modify to use STL containers
- **tracks**
  - implementation of the STAR track model
  - able to handle straight lines
- **utilities**
  - parser and input prompt

- **Compiles on:**
  - **HP:** aCC (v1.13); g++ (v2.8.1)
  - **Linux (Red-Hat 5.0):**: egcs (1.0.2)
- **Rotations for StThreeVector and StMatrix are coming...**

Concerns With DataBase Usage in TRS

- **Decouple the program from DB application**
  - Methods of Access are different
    - ASCII
    - Objectivity
    - etc...
  - Change **DB** mechanisms without rewriting code
  - Encapsulation
- **Avoid Multiple DB instances**
  - Make sure there is **NO WAY** to access 2 **different** geometry databases simultaneously
  - i.e. Printers and spooler

DataBase Decoupling and Encapsulation

- **User wants access to parameters from a Specified DataBase**
  - ...without Knowledge of how it is done

- **Define an Abstract Class that defines an INTERFACE**
  - Identical for ALL implementation
  - Allows flexibility for every Implementation
  - Bookkeeping is done with the Base Class
  - Minimal code modification when DB changes!!

Multiple Instances of a DataBase

Why Important?

- **DataBases and Initialization incur large overhead**
- **PROBLEM:How to Avoid Multiple Instances ...AND provide Global Access**
- **A Singleton Class Can be Used:**
  - Class is Responsible for its sole instance
  - Does Not Pollute Global Name Space

Implementation...

```
class StTpcSimpleGeometry : public StTpcGeometry {
public:
  StTpcGeometry* instance(string fileName)
    {
      if (!mInstance)
           mInstance = new StTpcSimpleGeometry(fileName);
      else
        ; // do nothing

      return mInstance;
    }

protected:
  StTpcSimpleGeometry(string fileName)
    {
      // do what needs to be done...
    }

private:
  static StTpcGeometry* mInstance;
};

// In my  program:

string geoFile(``example.conf'');
StTpcGeometry *geomdb =
        StTpcSimpleGeometry::instance(geoFile);

StInt rows = geomdb->numberOfRows();
```

see: E. Gamma, R. Helm, R. Johnson, J. Vlissides
Design Patterns: Elements of Reusable Object-Oriented Software

Coordinate Transforms

- **Requires 2 DataBase accesses:**
  - **TPC Geometry (pad plane)**
  - **Slow Control (drift velocity)**

```
StCoordinateTransform transformer(string);
// the Transform should not worry about
// where/how DB access happens!
// DO NOT pass a File --> Pass a dataBase

StCoordinateTransform
     transformer(StTpcGeometry*, StSlowControl*);
```

In TRS the main program (or StTrsManager) opens the ASCII file (or initializes Objectivity or Oracle or...) and the StTrsManager keeps the instance as a data member. It is passed to whatever needs it:

- **StTrsChargeSegment**
- **StTrsChargeTransporter**
- **StTrsSector**
- **...**

Transformations

There are 3 unique Coordinates

- **Raw**
  - sector, pad row, pad, time bucket
- **Local**
  - x, y, z
- **Global**
  - x, y, z

Use an overloaded Operator ()

```
StCoordinateTransform transformer(geoDb, SCDb);

transformer(raw,local);
transformer(raw,global);

// BUT...Must overload on dissimilar data type!!!
// Define 3 types of Coordinates as above

StTpcPadCoordinate(sector,padRow,
                        padNumber,TimeBucket);
StTpcLocalCoordinate(x,y,z);
StGlobalCoordinate(x,y,z);
```

Transformations

Access is only through () operators. For example:

```
transformer(raw,local);

// would call the appropriate
// private member function(s)
// For Example:

void StCoordinateTransform::
  operator(const StTpcPadCoordinate& a,
                      StTpcLocalCoordinate b)
{
  yLocal = yToRow(a);
  xLocal = xToPad(a);
  zLocal = zToTimeBucket(a);
  b = rotateToSector(StThreeVector<double>
                     (xLocal,yLocal,zLocal));
}
```

Pad Response Functions

- Simulate single tracks (cosmic test)
- Calculate charge induced from charge configuration on wires
- No assumption of signal shape
- Tune to an analytical (FAST) parameterization
- Tune to Cosmic rays...
- Large spread in prf...

Moving into a fixed framework with usable results coming out....