MIPP Offline Software Review

Jonathan M. Paley

02/07/05

Indiana University

Some Basics:

- Offline software is C++/Root based.
- Software is maintained using CVS & SRT.
- Raw data is converted to Root format using the *raw2root* program.
- Basic class for reconstruction and analysis: *JobCModule*
- "Reco" and "Ana" jobs are plugged into the analysis program, *anamipp* (as well as the Event Display: *evd*).
- Jobs are configurable using the *CfgConfig* class. Configuration may be done using XML (or the database?).

raw2root

> raw2root -o [output].root mipp[run].[subrun].raw

- Converts raw data into Root format
- Packages used:
  - *ConnectionMap*
    - electronics (crate, module, channel) <-> physical (plane,wire)
  - *EventDataModel*
    - overall MIPP data (tree) structure
  - *MippIo*
    - unpacks the raw data
  - *RawData*
    - defines the structure of the data (plane,wire or stick,pad,bucket)
  - *MippDatabase*
    - provides interface with MIPP postgres DB
  - *MippXML*
    - provides interface for parsing XML files

JobCModule

- Basic class for reconstruction and analysis
- Reco (Ana) jobs are configurable via the CfgConfig class
- Taken from part of the TPCRecoJP package:

```
class TPCR2DClusterFind : public JobCModule
{
public:
  TPCR2DClusterFind(const char* version);
  ~TPCR2DClusterFind();

  JobCResult Reco(EDMEventHandle& evt);
  JobCResult Ana(const EDMEventHandle& evt);

  void Update(const CfgConfig& c);

private:
  int fPedestal;
  int fDebug;
  int fPrintDebug;
  int fMinSeparation;
  int fMaxNumDig;

};
```

```
TPCR2DClusterFind::TPCR2DClusterFind
(const char* version) :
  JobCModule    ("TPCR2DClusterFind")
{
  // Do this first to load default config.
  this->SetCfgVersion(version);
}

void TPCR2DClusterFind::Update
(const CfgConfig& c)
{
  c("Debug").     Get(fDebug);
  c("PrintDebug").Get(fPrintDebug);
  c("fPed").      Get(fPedestal);
  c("MinSep").    Get(fMinSeparation);
  c("MaxNumDig"). Get(fMaxNumDig);
};
```

```
JobCResult TPCR2DClusterFind::Reco(EDMEventHandle& evt)
{
  int result, evtnum;

  TDirectory* dir = gDirectory;

  // list of raw digits to get from event
  ConstDigVec digitList;
  ConstDigVec rdigitList[GTPCConst::kNPadRow];

  // list of 2D clusters to be formed from the raw digits
  Clus2DVec cluster2dList[GTPCConst::kNPadRow];

  // get a list of digits from the event
  digitList.clear();
  try {
    evt.Raw().Get("./tpc", digitList);
  }
  catch (EDMException e) {
    std::cerr << "Failed while loading TPC data ...\n";
    return JobCModule::kFailed;
  }

  if ((int)digitList.size() > fMaxNumDig) return JobCModule::kFailed;

  { do stuff...}

  dir->cd();

  return JobCModule::kPassed;
}
```