Visualization

Objects (Plottables) are presented to the User via their Representations, which are shown in Scenes.

Representations can be chained.

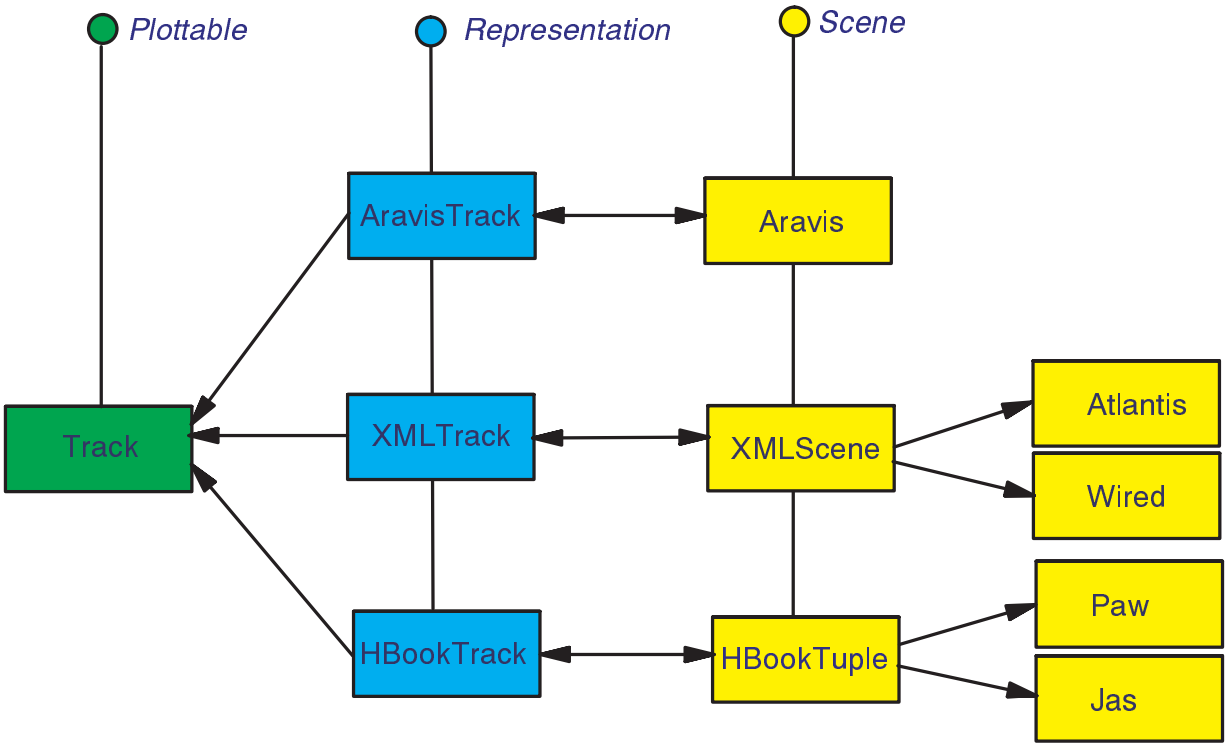
The mechanism is transparent to the user.

http://atlasinfo.cern.ch/Atlas/GROUPS/GRAPHICS/

standard Atlas Graphics Architecture fully supported in Paso
representations of an object:
* 3D object
* 2D object
* XML element
* textual description
* histogram
* n-tuple
* ...

see detailed documentation on the Web (Paso, Aravis,...)

Example



simplified example
abstract interfaces
we are on migration to upgraded Design (Lassi)

Scenes in Paso

Other Scenes are coming.

- one-to-one representations:
  - Event Display:
    - Aravis - integrated
    - Atlantis - via XMLScene
    - AVRML - for VRML Browsers
    - Wired - via XMLScene, rmi, Corba
- many-to-one representations:
  - Statistics
    - HBookTuple - for Paw, Jas
    - AHTL - for LHC++
- one-to-many representations:
  - Detector Display
    - Aravis - integrated
    - AVRML - for VRML Browsers
- misc:
  - AsciiText
  - XMLScene

currently only Aravis is integrated into Paso, others use files to communicate

Plottables in Paso

Creation of Representations for new Plottables is highly automatised, but SubSystems should define which Objects to visualize and how.

- TruthEvent
  - TruthTrack
- InDetEvent
  - SiDigit
  - TRTDigit
  - SiDetector
  - TRTDetector
  - StripCluster
  - SpacePoint (soon)
- any other which SubSystem identifies

End Programmer

```
SceneList sl;                    // create control
XMLScene xml("MyFile");          // open XML file
AVRML vrml("MyFile");            // open VRML file
HBookTuple hbook("MyFile");      // open HBook file
Aravis aravis();                 // open Aravis window
sl.add(xml);                     // register xml
sl.add(vrml);                    // register vrml
sl.add(hbook);                   // register hbook
sl.add(aravis);                  // register aravis

// .... create or get TruthTrack

sl.show(TruthTrack);             // send TruthTrack to xml, vrml, hbook
                                 // show TruthTrack on aravis
```

then look
at MyFile.xml with WIRED or Atlantis
at MyFile.wrl with VRWeb
at MyFile.hbook with Paw or Jas

very simple

Plottable Developer

```
$ createPlottableModel TruthTrack      # create sceletons for all classes needed
$                                      # for TruthTrack visualisation
$                                      # they will compile and link, but without
$                                      # any real nontrivial efect
$
$ nedit VRMLTruthTrack.cxx             # define VRML behaviour
$ nedit HBTupleTruthTrack.cxx          # define HBook behaviour
```

```
// pTrack is available
// ...
aVertex = pTrack->vertex();       // get Vertex
HepPoint3D pos(aVertex);          // create position
HepPoint3D dir(pTrack->p());      // create direction
Line aLine(pos, dir);             // construct line
add(aLine);                       // add line
// ...
```

```
// pTrack is available
// ...
tuple->column("px", pTrack->p_x(), 0);  // add px
tuple->column("py", pTrack->p_y(), 0);  // add py
tuple->column("pz", pTrack->p_z(), 0);  // add pz
// ...
```


creation of all structures needed to add any new Plottable is done by the Wizard

the SubSystem should defined the content (how they want to see data)

XML

- atlasevent.dtd for Event Data
- AGDD.dtd for Detector Data
- PlotML.dtd for Statistics Data

http://atlasinfo.cern.ch/Atlas/GROUPS/GRAPHICS/Texts/Documentation/XMLScene/XMLinAtlas.html

XML is becoming widely used: AGDD, Wired, Atlantis, Jas,...
informations about Tools on the Web