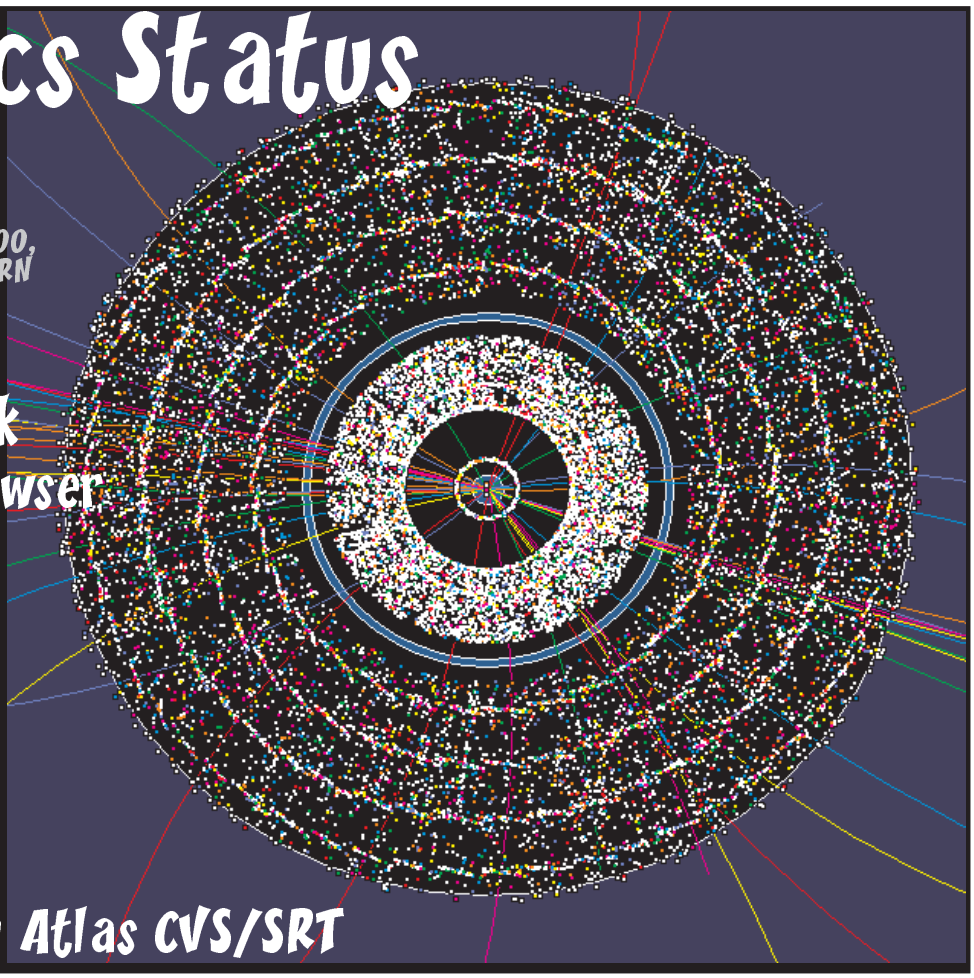
Graphics Status

J.Hrivnac, 1 September 00,
Atlas Software Week, CERN



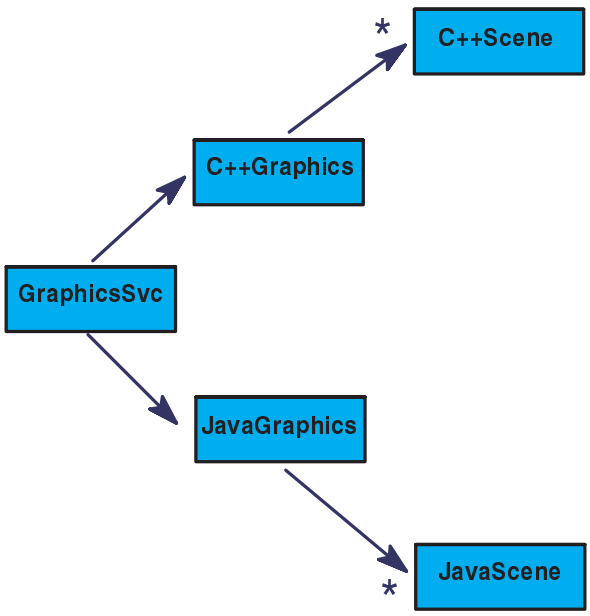
- Framework
- Object Browser
- Aravis
- WIRED
- Atlantis
- GraXML
- JAS
- JACO
- Graphics in Atlas CVS/SRT

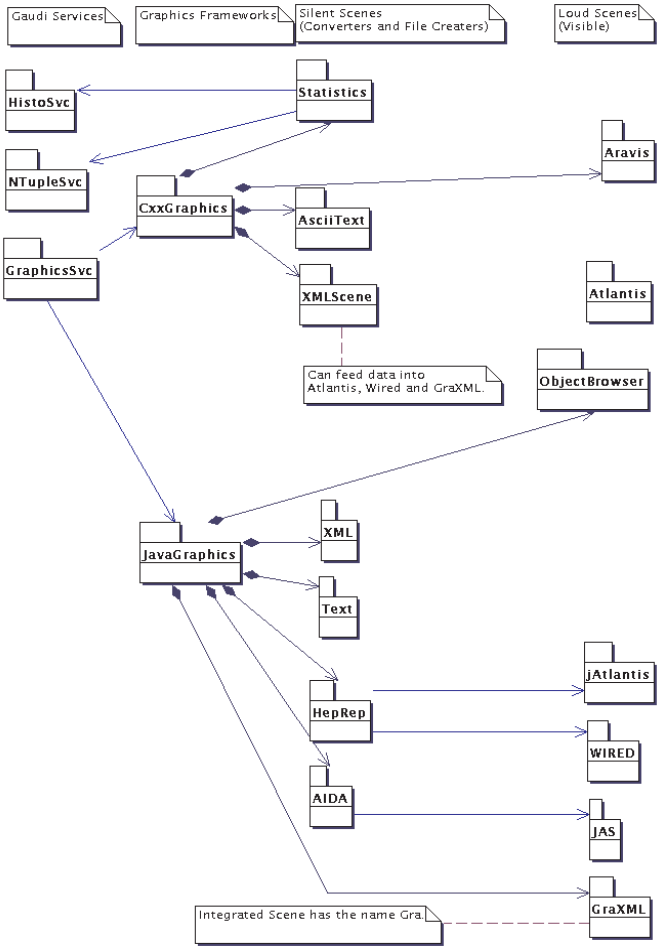
Framework

- Both C++ and Java Frameworks fully functional
  - same architecture (Plottable-Model-Rep-Scene)
  - Java Framework much more functional
  - C++ Framework frozen
- Graphics in Athena:
  - as Algorithm next week
  - as Service after Athena available in release
  - easy port of "Paso Service"
  - two Services: Java and C++
- Java-C++ Interface:
  - using JACO
  - work has just started (Mark + Tony)
  - initial development in the Paso Framework
  - will give also Java Scripting as by-product

- Java Framework functional but not yet coupled with Athena
- Algorithm will be implemented as extension of Tutorial
- Algorithm will be implemented as reply to several requests
- initialy, only C++ Framework will be interfaced
- Java scripting (using BeanShell,...) will have access only to data interfaced for Graphics (Event + DetDescr)

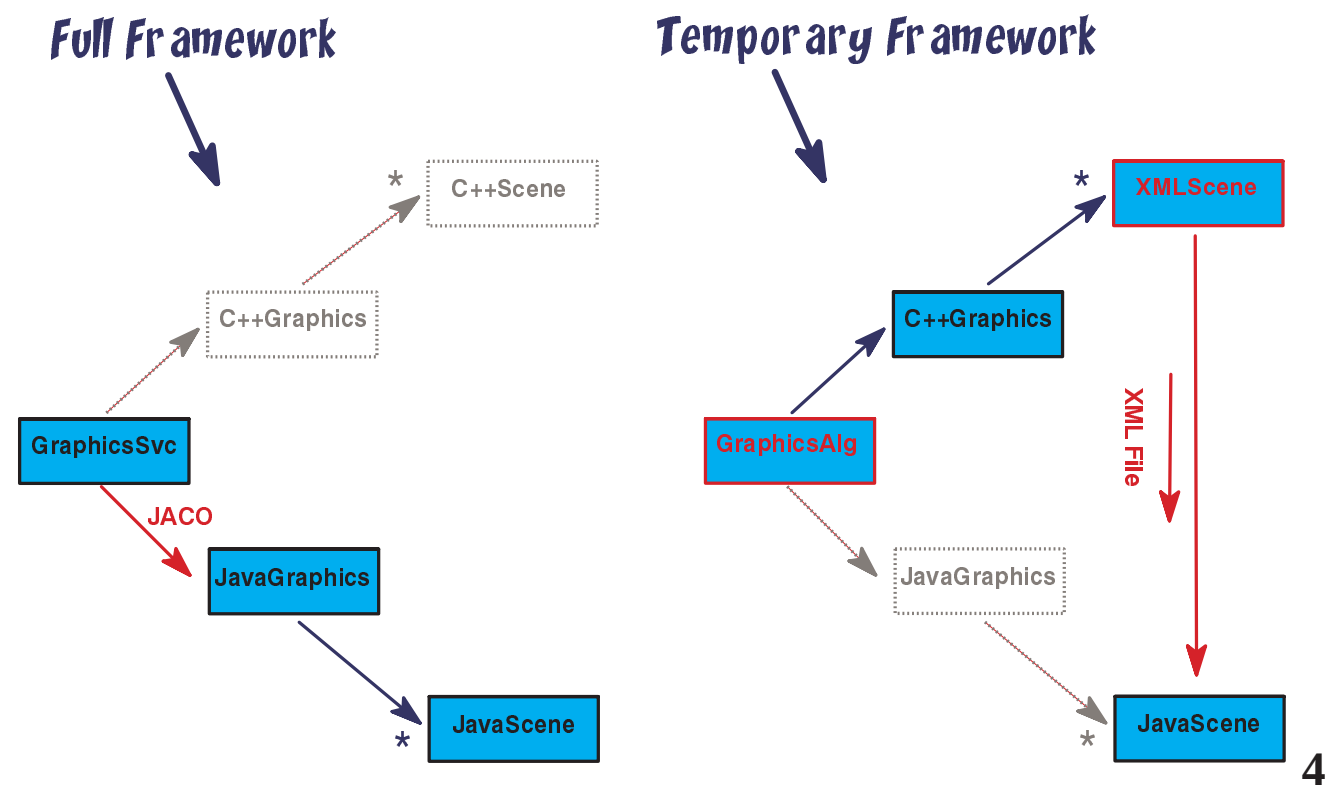
Domain Decomposition





- Statistics will use standard Gaudi / AIDA Services

Domain Decomposition



- untill full Java Chain is available, XML is used as bridge for data

Reps

- ID:
  - Barrel: OK
  - EndCap: nearly OK
- MU:
  - Barrel: OK
  - EndCaps: disagreement between Digit and Detector
- Calo:
  - very naive
- Truth:
  - Helix (=> OK for ID, not good for MU)
- Reco:
  - nothing yet

Context Sensitive Actions (CallBack) under development.

Scenes (C++)

- AsciiText
- XMLScene:
  - provisional bridge to Java Framework
- Aravis ← See David' talk
- Atlantis ← See Gary' talk

Scenes (Java)

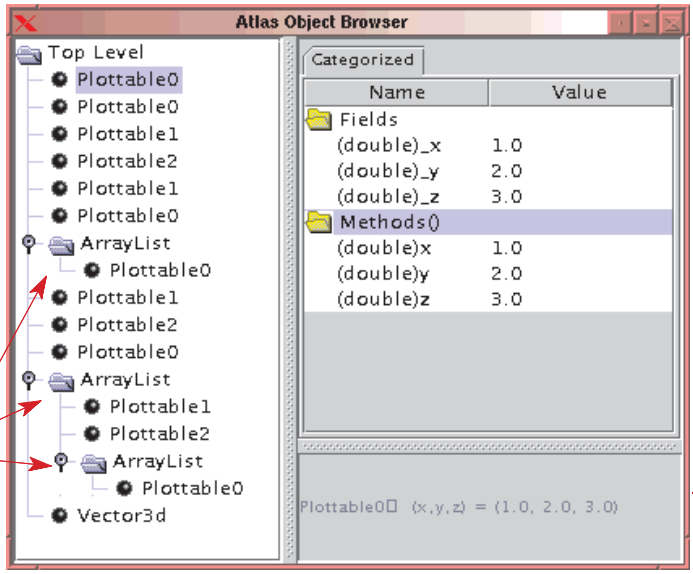
- Test:
  - provides testing of the Graphics Framework
- Text :
  - dump into text file
  - has default Reps (via Reflection)
- XML:
  - dump into XML file
  - has default Reps (via Reflection)
- AIDA:
  - histograms,...
  - dump into:
    - to text
    - to XML
    - to JAS

Scenes (Java)

- Object Browser (see below)
- GraXML ← See Julius' talk
- jAtlantis ← See Gary' talk
- WIRED ← See Mark' talk
- JAS ← See Tony' talk

Object Browser

- Originaly from Luiz Cunha & Lassi Tuura
- Ported as standard Scene



Standard Collections automaticaly expanded

Fields and Methods extreacted by Introspection (may be dangerous)

toString() description

- may be included in FreeHep common Java distribution
- may be used by JAS
- should it give private properties ?

Object Browser

- Introspection used to define standard behaviour, which can be changed by Reps
- Legacy C++ Objects will be accessed by **JACO**
- **Context Sensitive Actions** will be added to:
  - send Plottables to other Scenes (possibly using drag&drop)
  - call Objects' proper methods (may be dressed by Models or Reps)

Aravis

- migrated from Motif to Qt (R.&D.Candlins)
- currently the same fuctionality & UI
- much easier to extend functionality

WIRED

(M.Donszelmann, Ch.Loomis, J.Perl, T.Johnson)

- migrated to HEPRep (HEP standard for Event Displays)
- implementing as JAS plugin
- new graphics engine for 2D
- adding Java3D

WIRED / FreeHEP

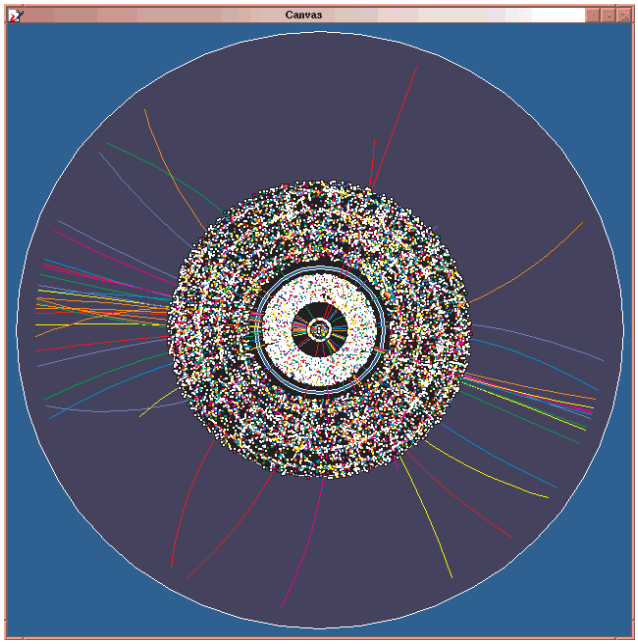
- library of common HEP & non-HEP Java classes
- together with Colt & WIRED & JAS equivalent mission as CERNLIB/F77 or Anaphe/C++ (or other similar projects)
- Atlas contributes

Atlantis

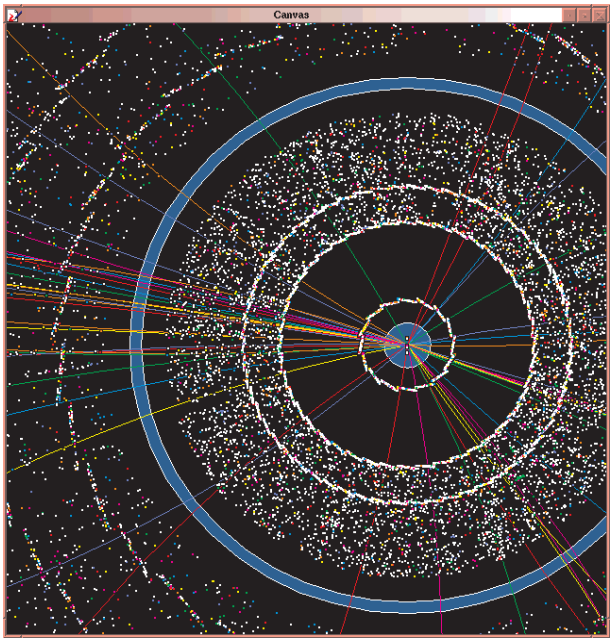
(H.Drevermann)

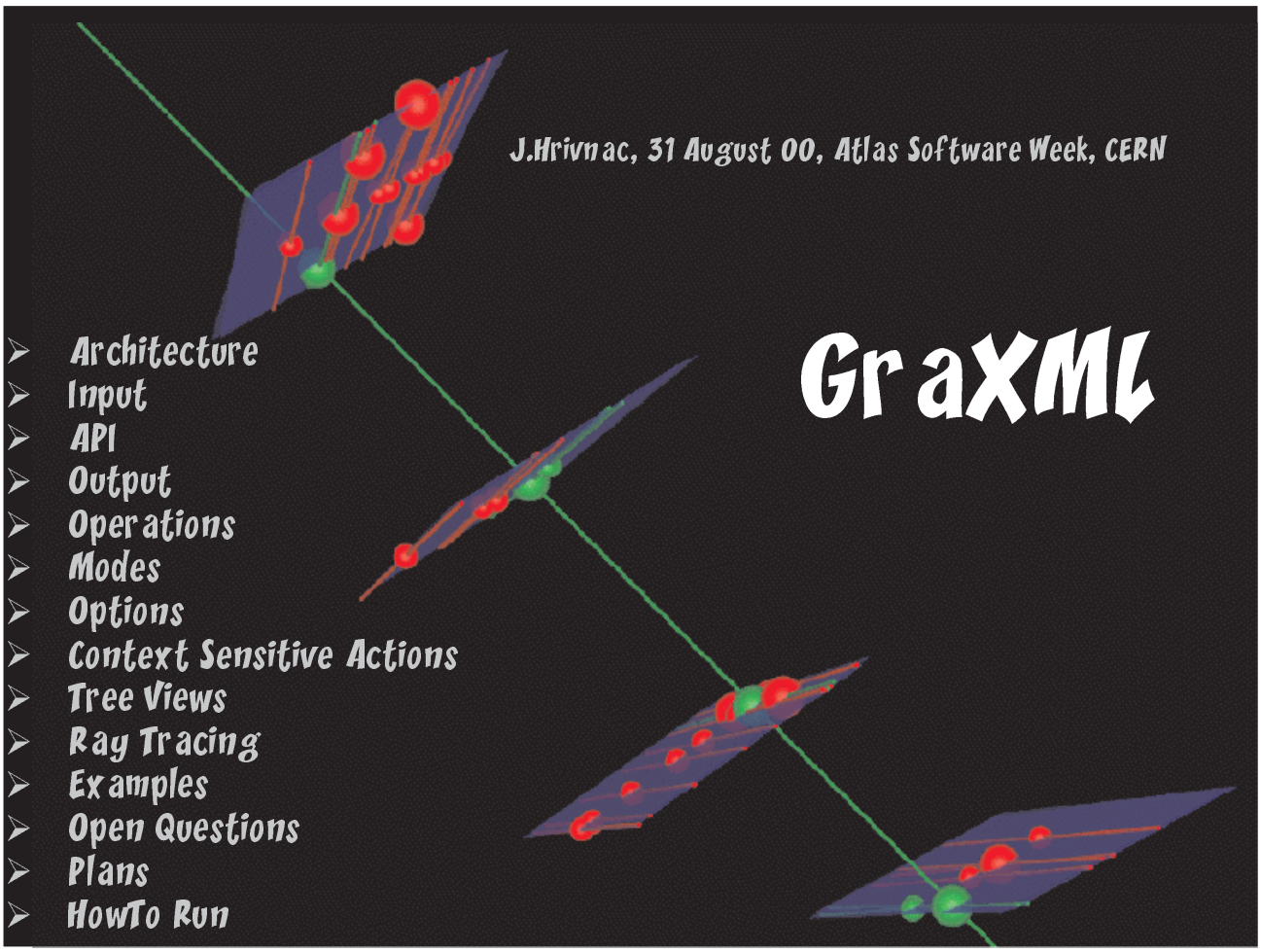
- being reimplemented in Java
  - graphics engine (G.Taylor)
  - GUI (P.Klok)

☒Full



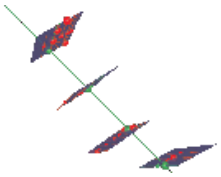
☒Detail





15

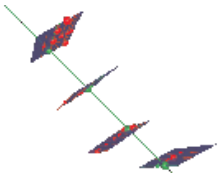
- big step since last meeting



Architecture

- Uses standard Plottable-Model-Rep Design
- Runs both as standalone and as a Scene
- Uses Java 1.3 + Java3D 1.2 (+ OpenGL)
- Uses X3D for VRML/XML output

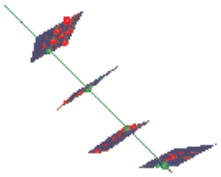
- a bit modified standard Design
- migrated from X3D-based into Java3D-based
- Java3D makes interactiveness easy
- OpenGL => profits from videocard acceleration (like VooDoo)
- the same performance as OpenInventor



Input

- **AGDD XML files version 4.2** (missing stacks and binary operations)
- **AtlasEvent XML file**

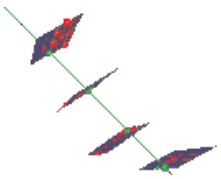
- can read some aditions to AGDD 4.2 (like skewed trd)
- stacks will be implemented in October by Stan
- several XML files (of both types) can be used together



API

- Standard Atlas Graphics Design (in Java)
- In future as Graphics Engine in WIRED

- will become part of WIRED-JAS-jAtlantis Framework

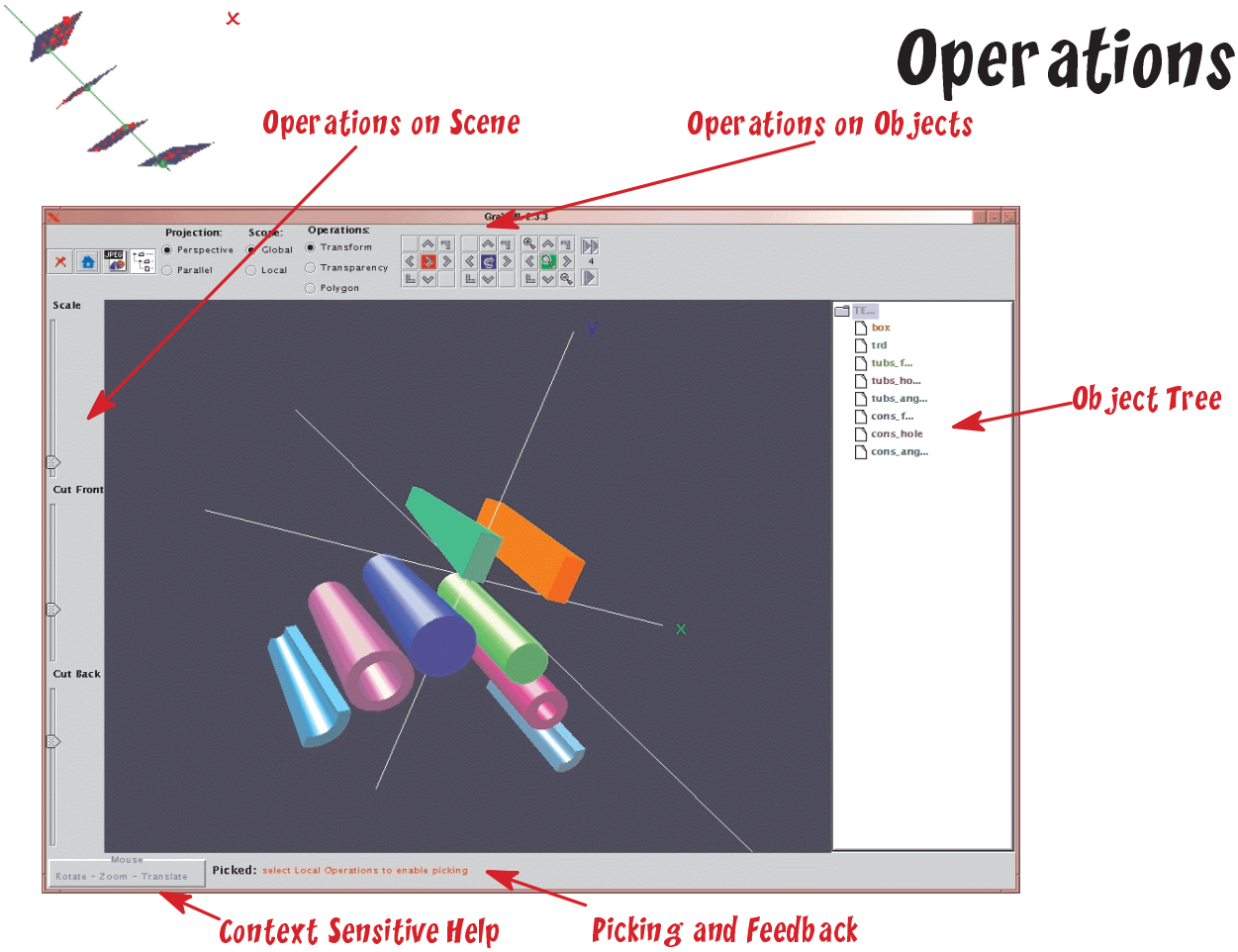


Output

- VRML97 (=> into standard VRML browsers)
- X3D (draft standard)
- JPG
- Soon PostScript

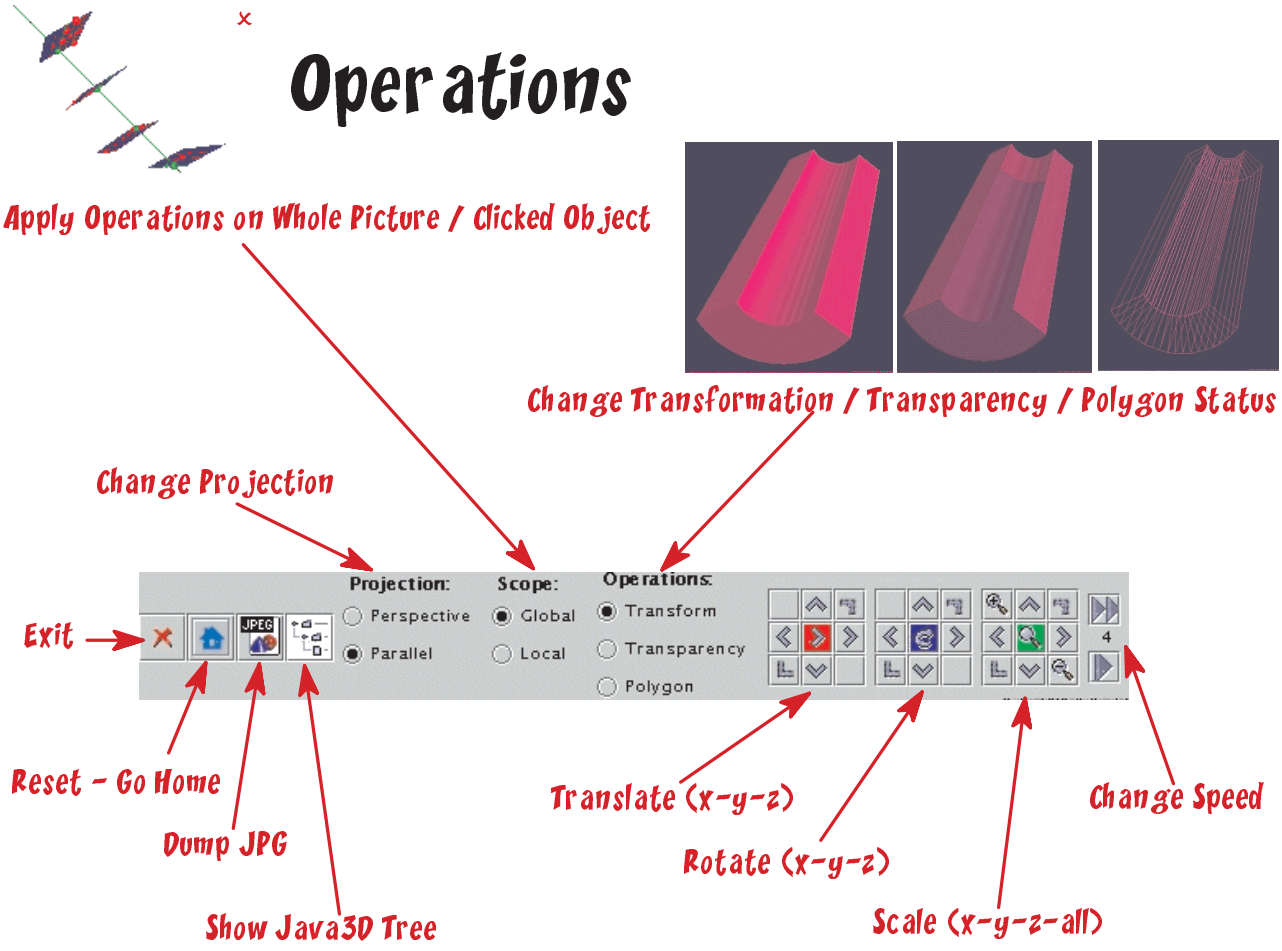
- PostScript will use FreeHep (JAS, WIRED,...) implementation

Operations



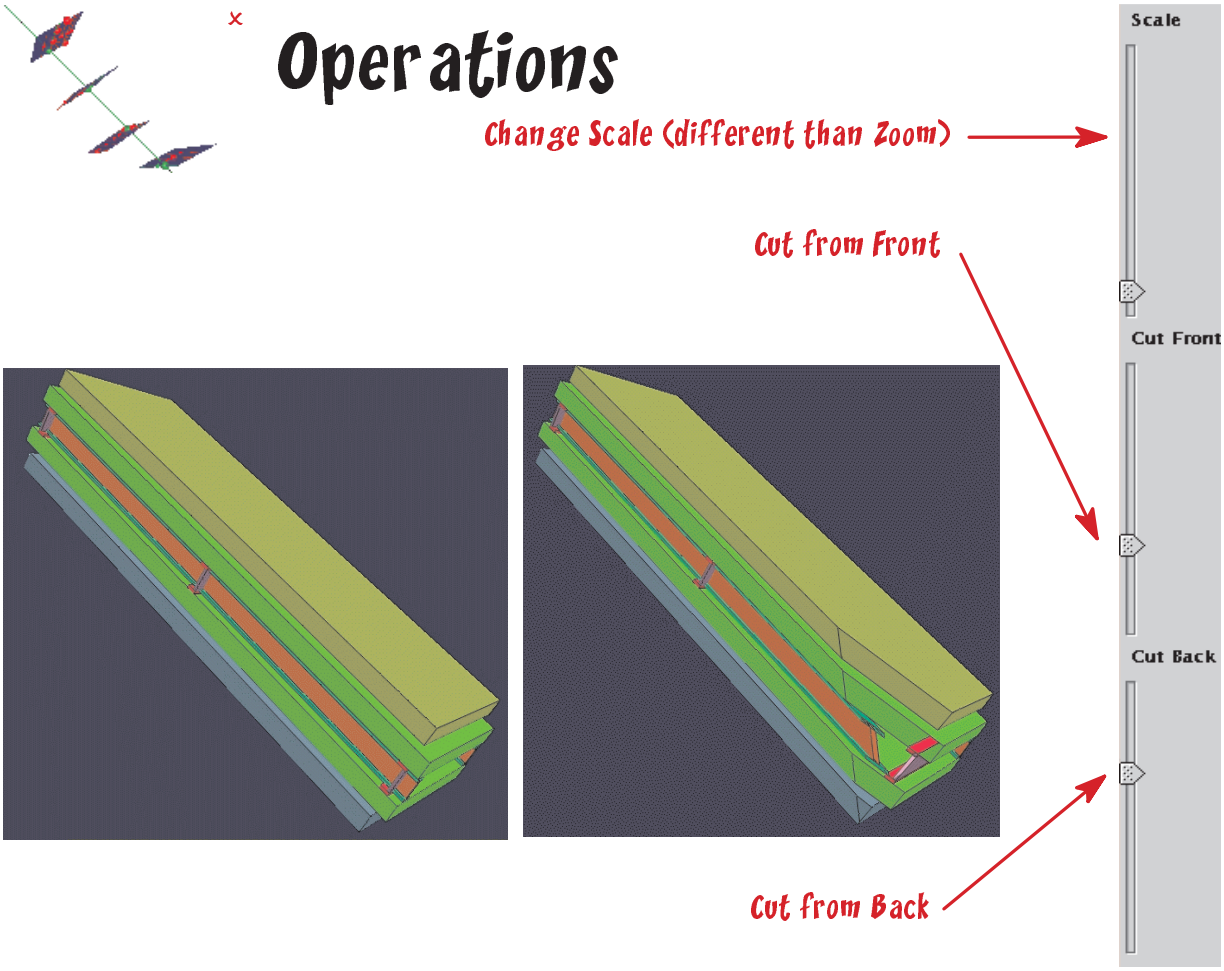
- 3D Objects will become part of FreeHep library
- GUI is not yet stable, it is under rapid development
- Operations on Scene are much faster

Operations

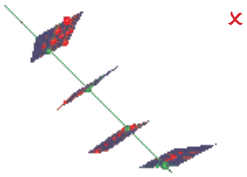


- will be improved
- Show Java3D Tree mostly for debugging

Operations



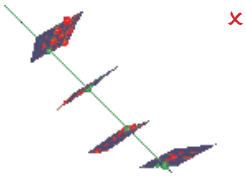
- Scaling much faster than Zooming
- Scaling to be used in Parallel Projection



Operations

- Transformations (Translate-Rotate-Zoom):
  - Mouse
  - Keybord
  - Menu

- all three way behave in different ways
- Menu based Transormations for big Scenes

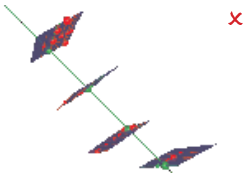


Modes

- Interactive [default]: -i | -n
  - Access to per-Object features
  - Slower & bigger
- Optimised (compressed): -o
  - Equivalent Objects as references
  - Faster & Smaller
  - Objects can't be switched off (as they have hidden references)
- Posponed [default]: -p | -q
  - Switching Objects On/Off
- Browser: -b [vrml|view|xml|vrml|...]
  - External output or browser

Selectable at start-time.

- the way how the SceneGraph is build and shown

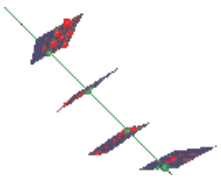


Options

- What to display:
  - Objects contributing into the SceneGraph
- How to display:
  - Representation to use (asLine, asBox, ...)
  - Cuts, colors, dimensions,... to use
- How to render:
  - Wireframe, Outline, Antialiasing,...
- How to optimise SceneGraph

Currently selectable at compile-time,
will use GUI / XML Options Files / Command-line.

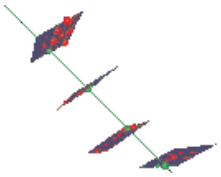
- large amount of Options



Context Sensitive Actions

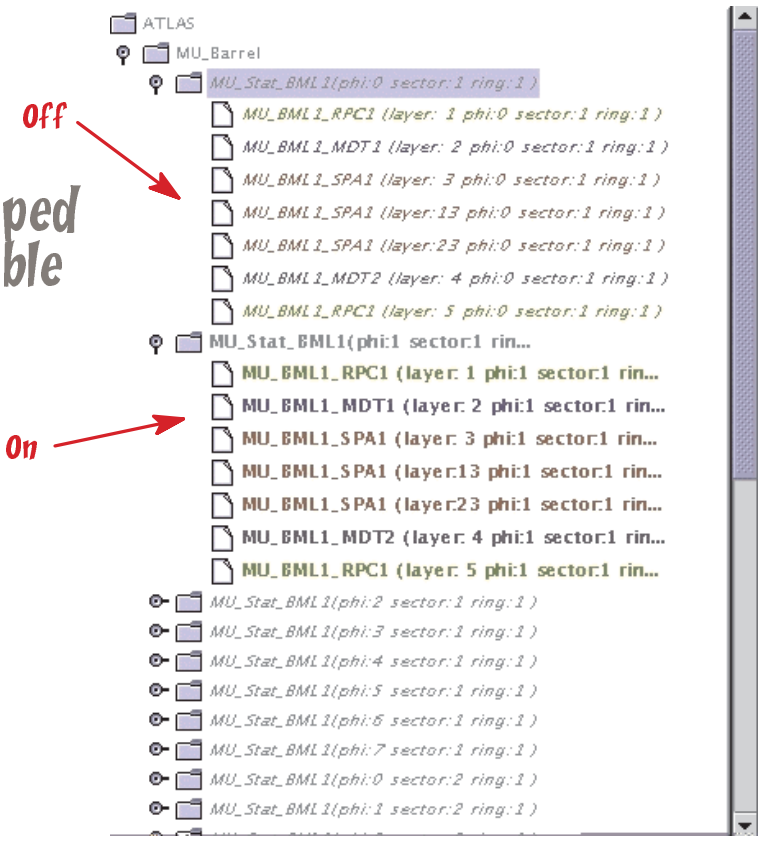
- 3D Rep Objects are alive and carry references to their Plottable origins
- Context sensitive Popup Menu
- Will use **JACO** access to C++
- Authors of original objects should define context sensitive actions (via Model)

- not yet fully implemented as meaningless without access to real Objects

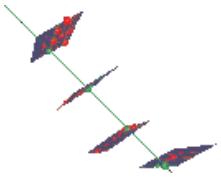


Tree View

- Tree of element maped into Tree of breakable connections
- Allows switching On/Off
- Uses identifiers
- Uses object' colors
- Can allow other operations



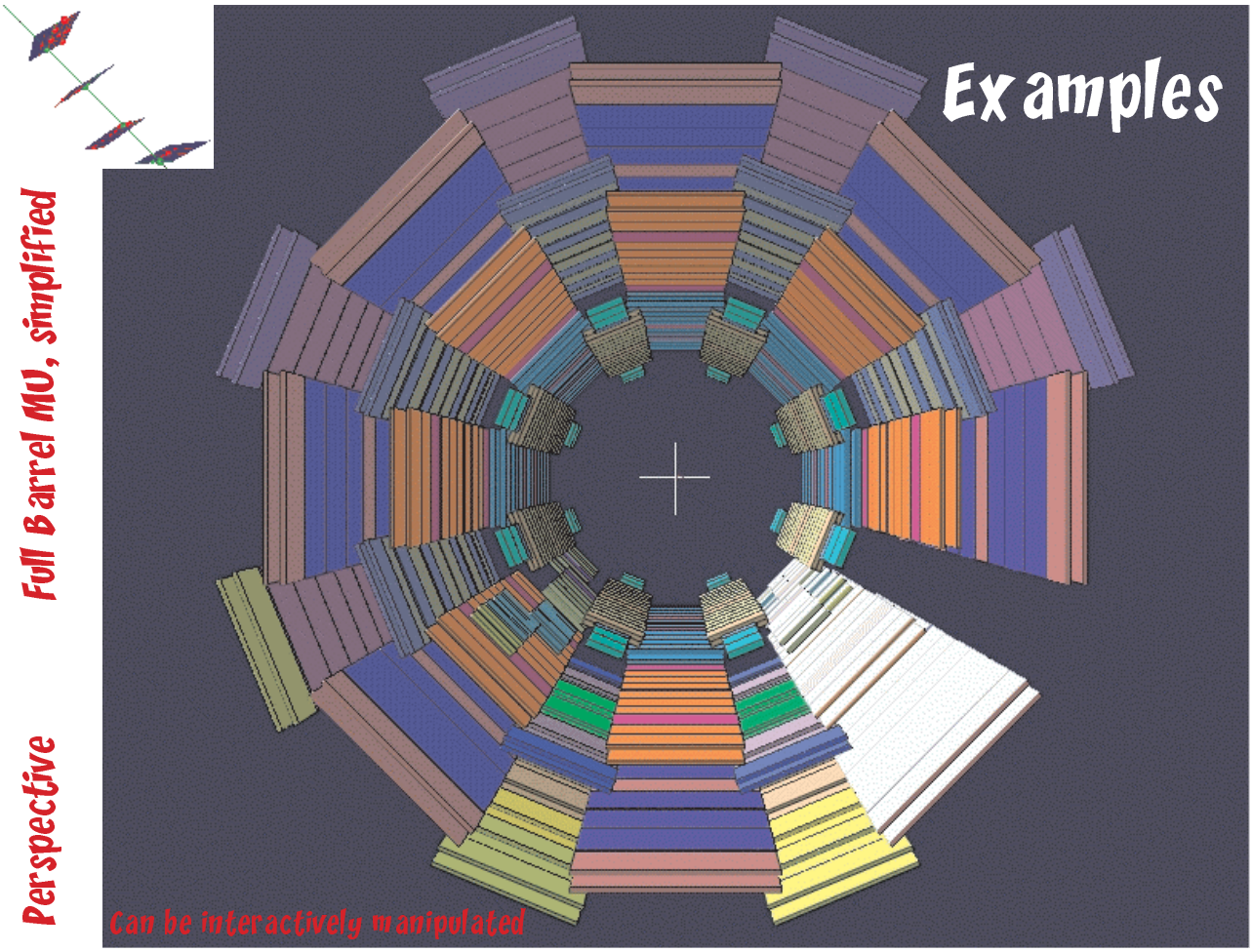
- identifiers not yet fully debugged
- not yet implemented for AtlasEvent (will be done within days)
- quite tricky Design



Ray Tracing

- Photo-quality output
- Java3D->VRML->PovRay
- Not part of the Framework

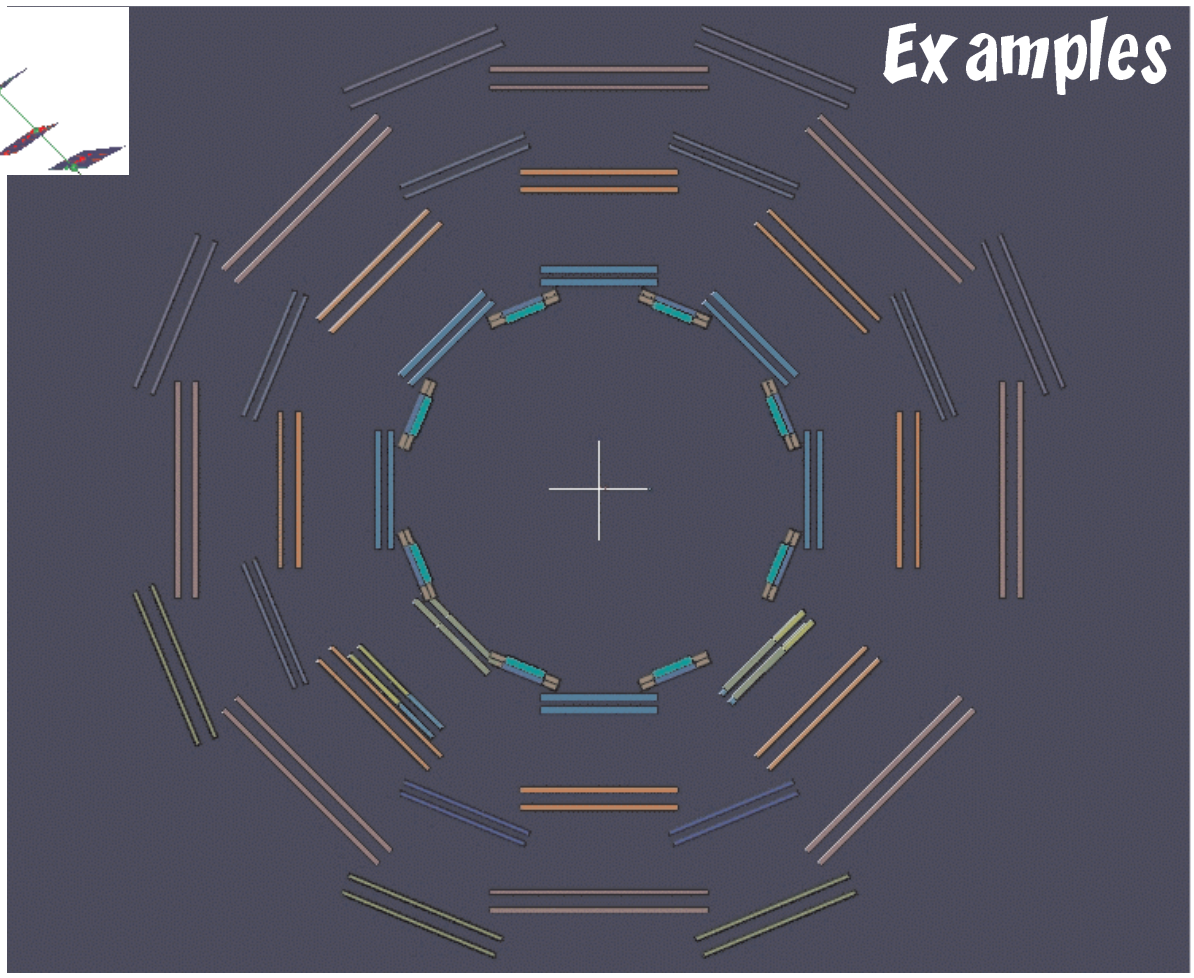
- can do shades,...
- slow
- will help Users on demand



- only MDT without spacers (to be used interactively)
- right bottom is reflection
- the missing pieces are not yet in the XML file
- colors are assigned per equivalent Object (same AGDD name,...)

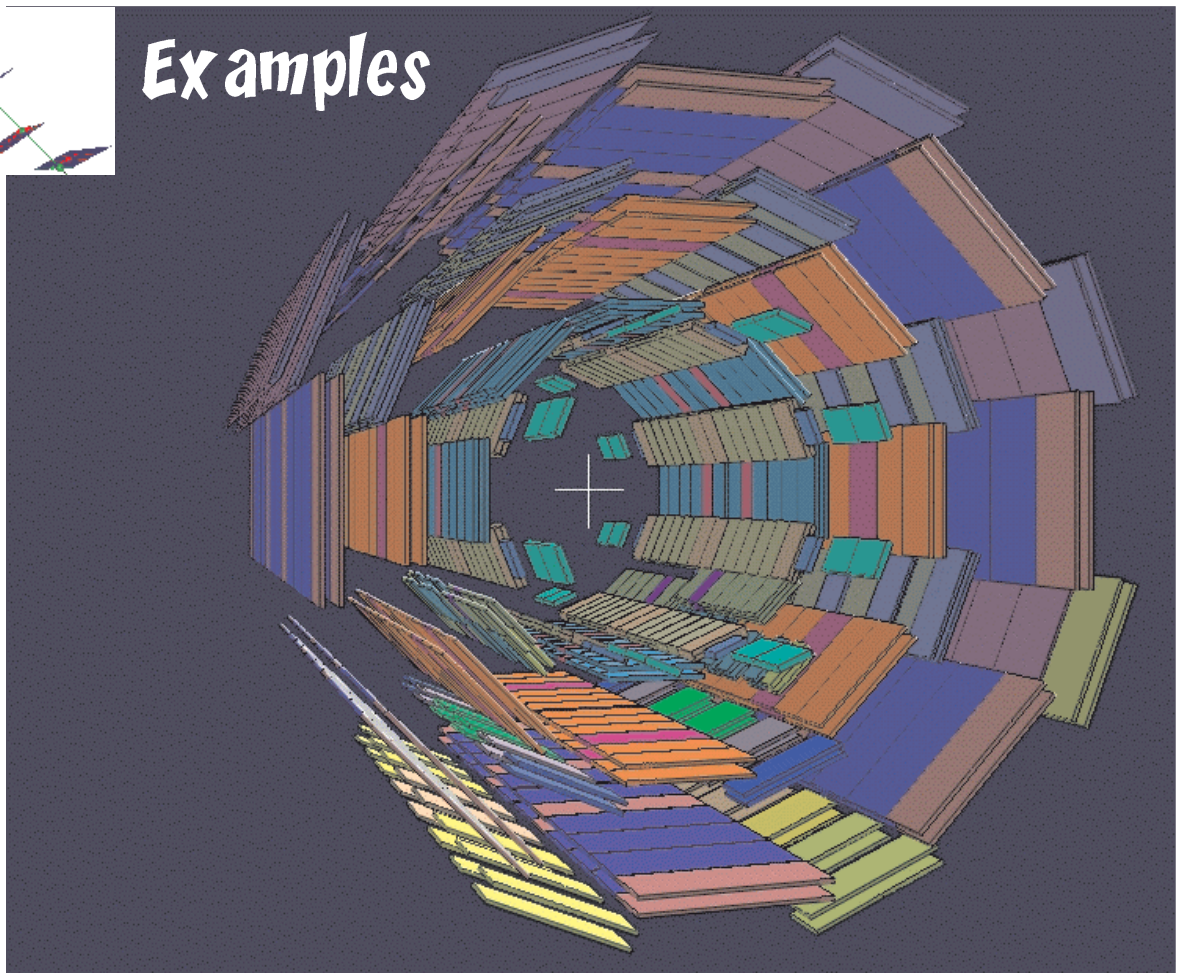
Examples

Full Barrel MU, simplified



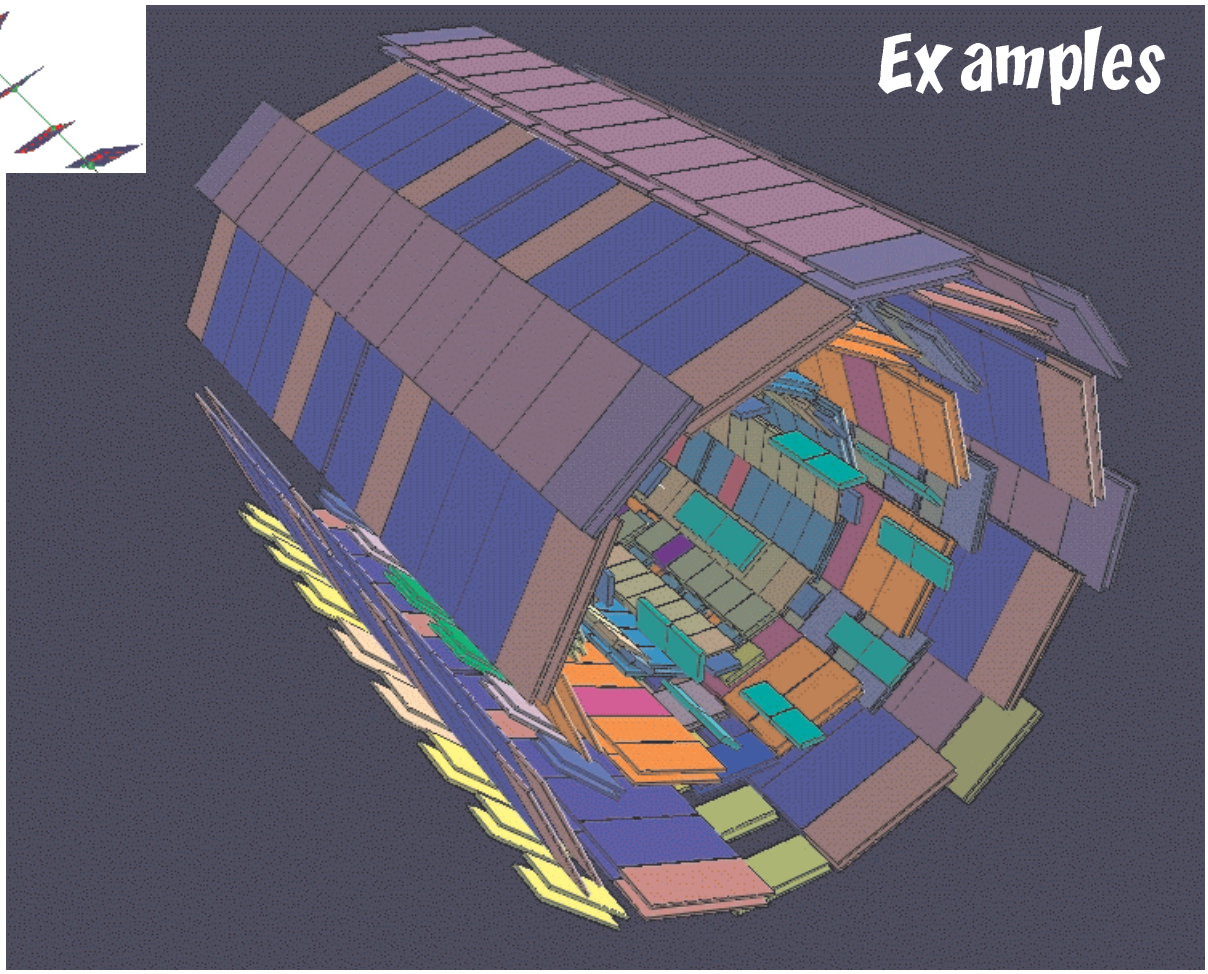
Parallel

Examples



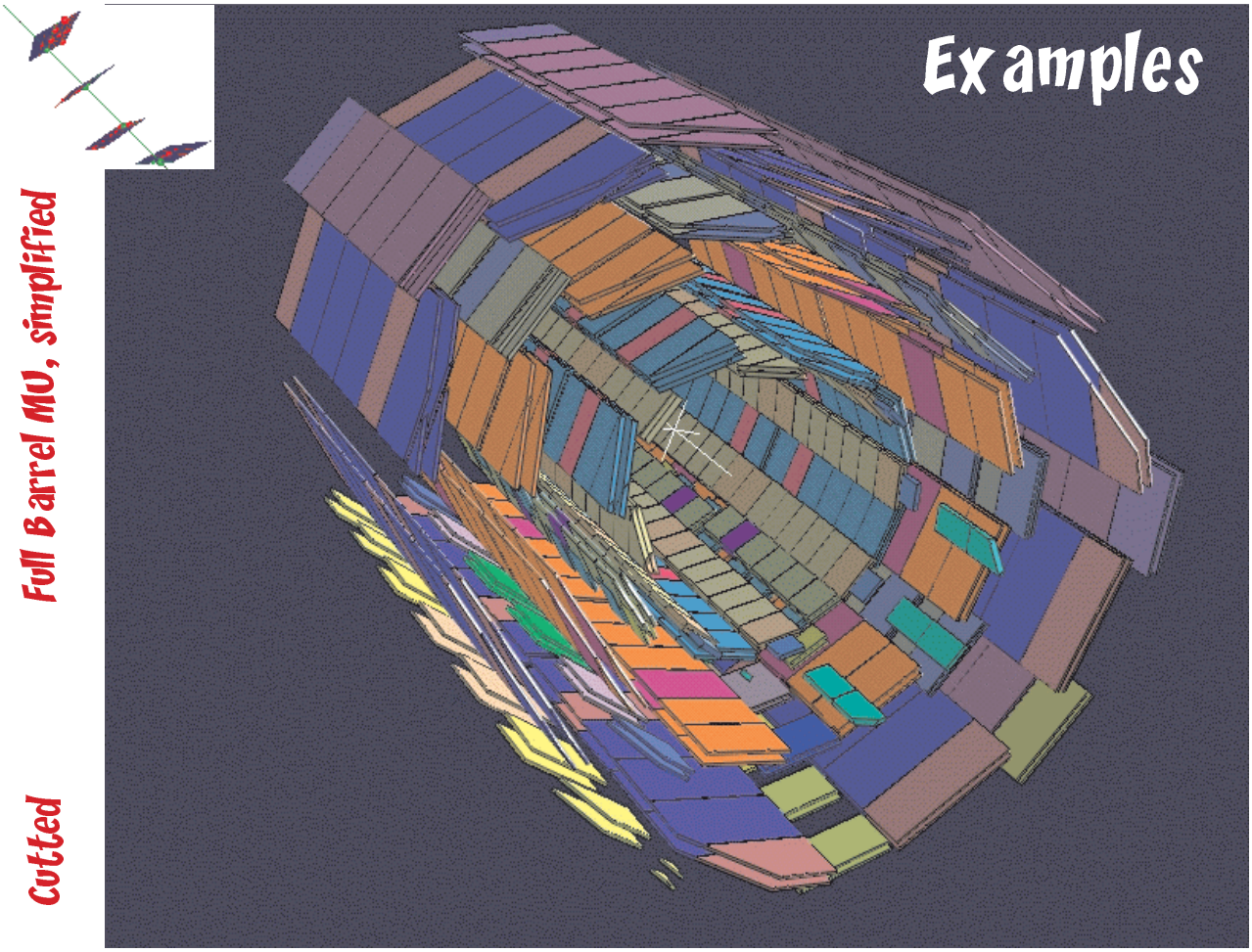
Full Barrel MU, simplified

Examples

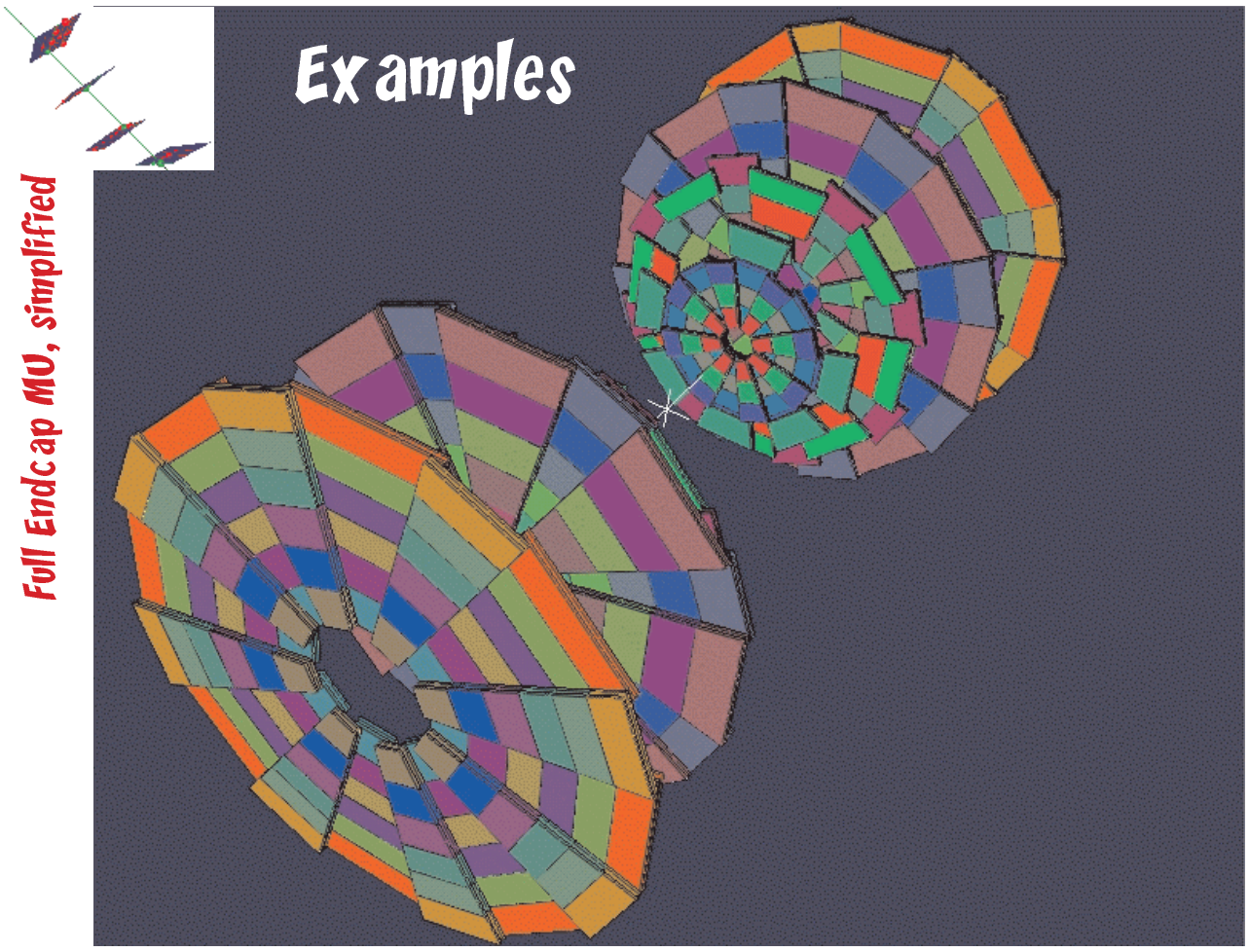


Full Barrel MU, simplified

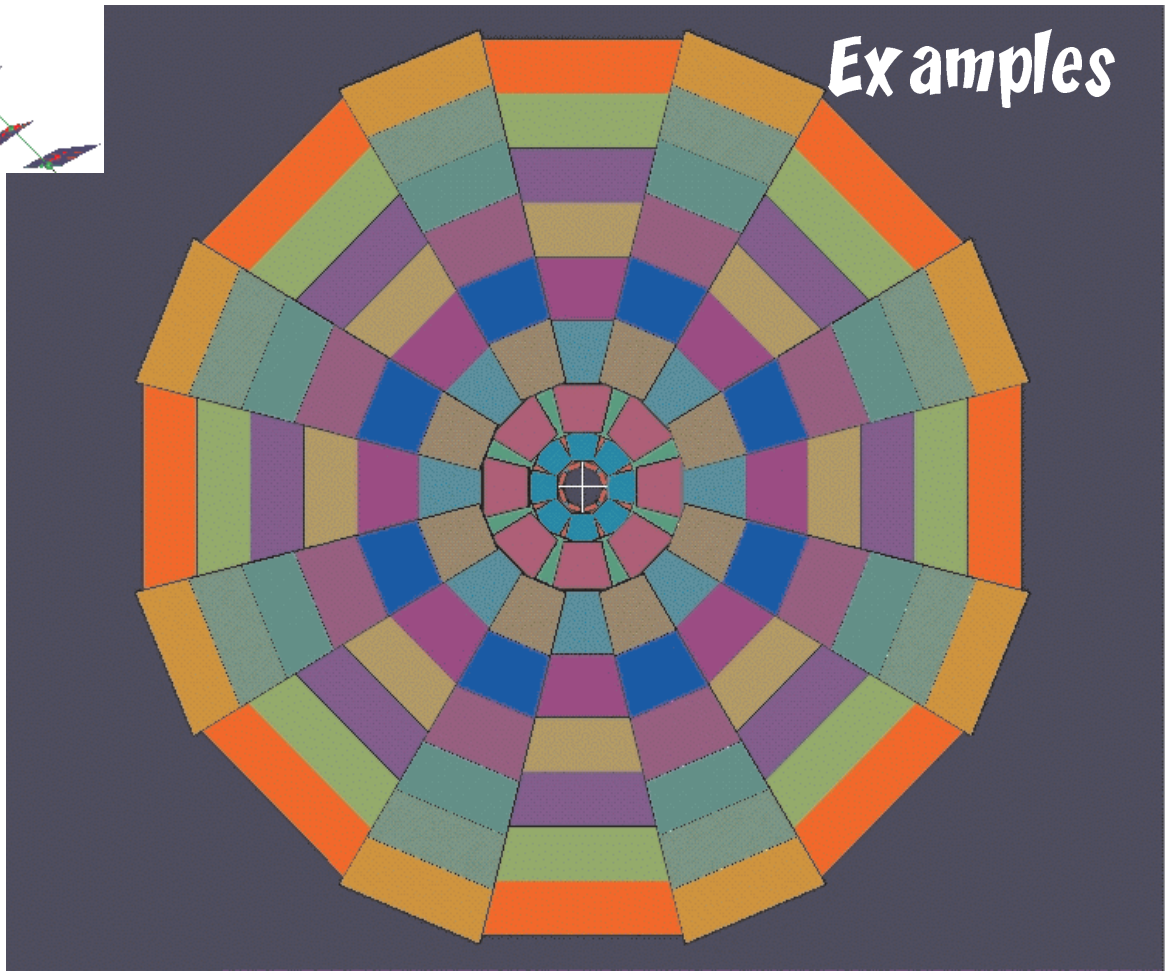
UnCutted

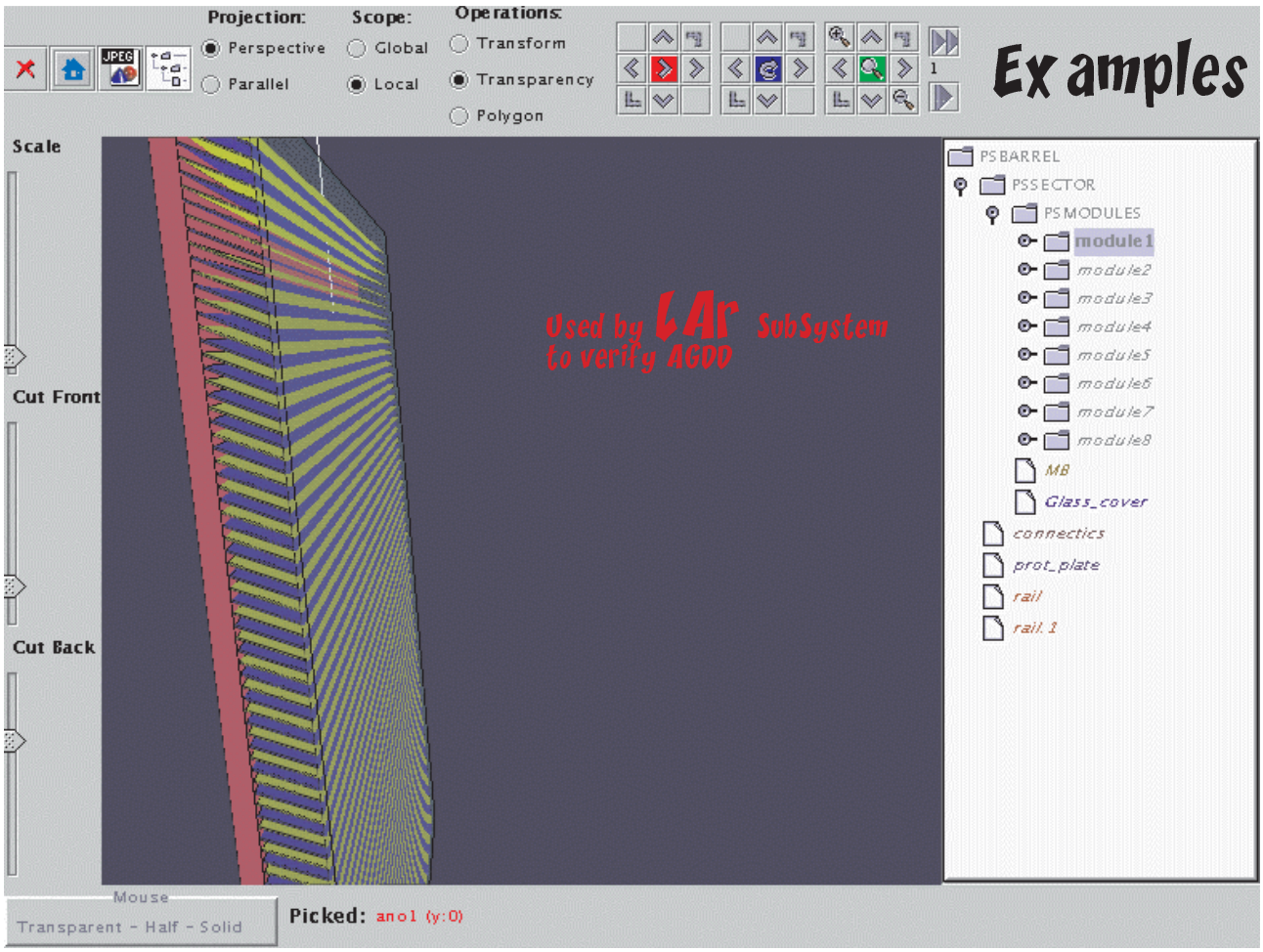


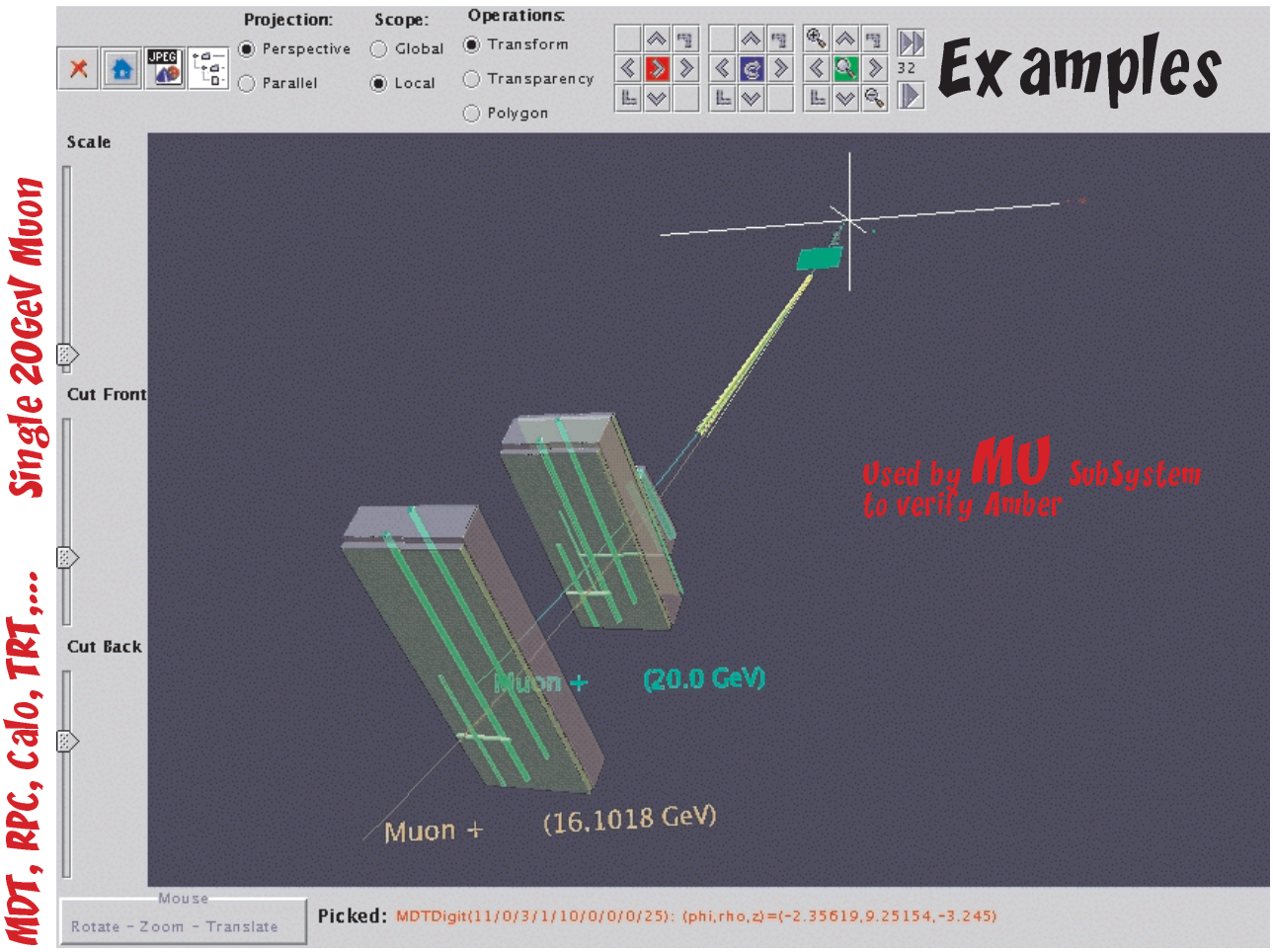
- cut allows to see what's inside
- cut is fast
- there're both front and back cuts



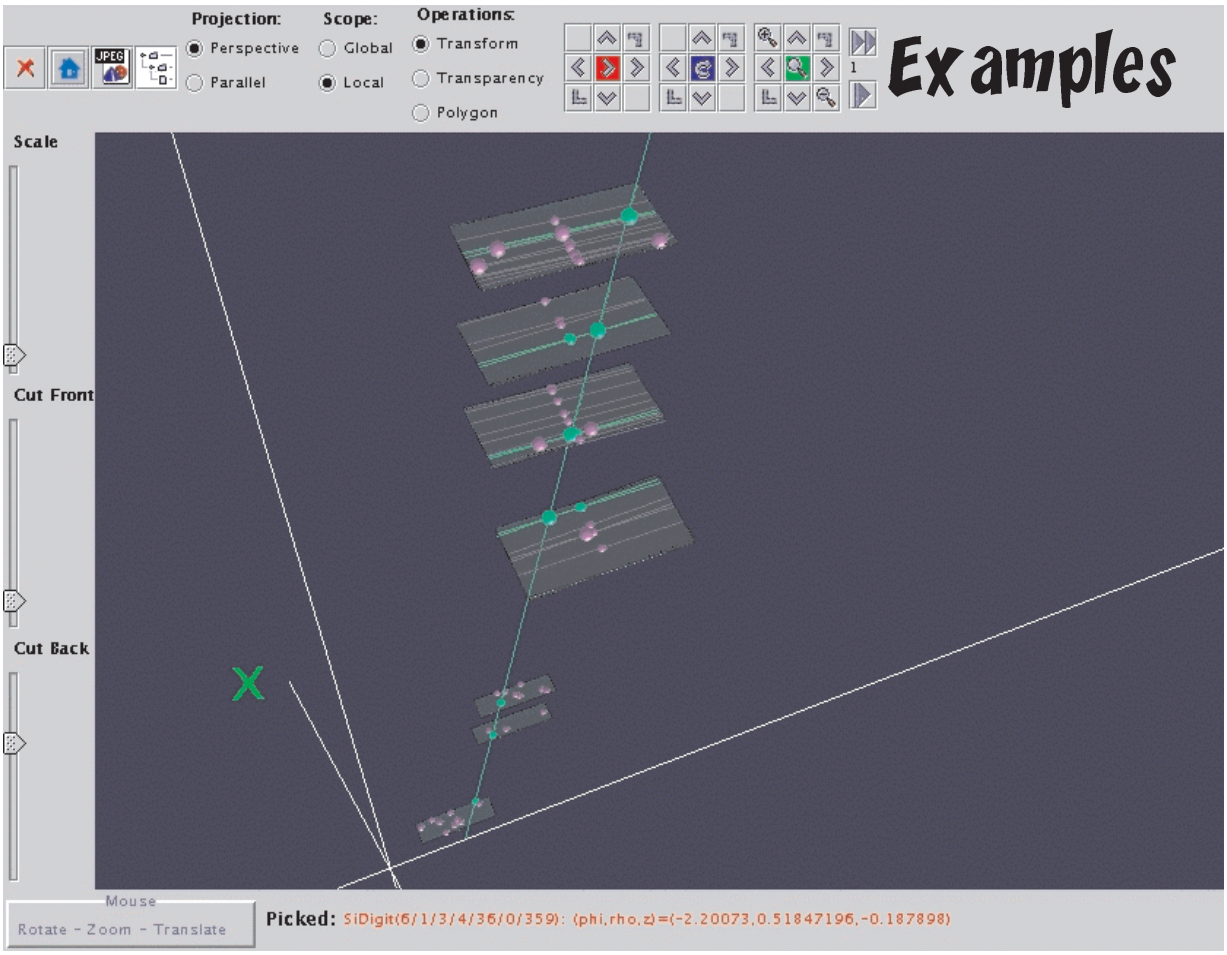
Full Endcap MU, simplified


Examples
Full Endcap MU, simplified


Projection:
Perspective
Parallel
Scope:
Global
Local
Operations:
Transform
Transparency
Polygon
Examples
Scale
Cut Front
Cut Back
Used by LAr SubSystem
to verify AGDD
PSBARREL
PSSECTOR
PSMODULES
module1
module2
module3
module4
module5
module6
module7
module8
MB
Glass_cover
connectics
prot_plate
rail
rail.1
Mouse
Transparent - Half - Solid
Picked: ano1 (y:0)



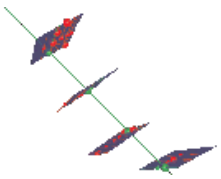
- MDTDigit clicked
- all (incl. Local) Operations applicable



Single 20GeV Muon

SCT, Pixels,...

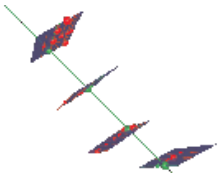
- SiDigit clicked



Open Questions

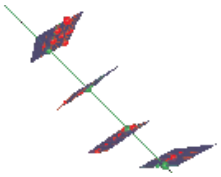
- Which Operations and Options to Expose ?
  - Almost everything is implemented, but it would be confusing to give access to all features.
- How to use Colors ?
  - Currently random colors are used for equivalent elements. Should it use color per Material ?
- Which default Representations to Choose ?
  - Mostly relevant for AtlasEvent. asPoint x asLine x asCylinder,... Also default dimensions. SubSystems should give feedback.
- How to balance Interactivity vs. Speed & Memory tradeoff ?
- Expose command line ?
- Support Stereo ? Other visual features ?

- now, it's important to choose what and how to use
- tradeoff between functionality vs. CPU and Memory consumptions and complexity (for Users)



Plans

- Improve Design, Refactor
- Add missing functionality (AGDD stacks)
- Add missing elements (mostly AtlasEvent)
- Remove unneeded functionality
- Fix bugs
- Improve UserFriendlines
- Improve Speed and Memory requirements
- Interface with WIRED (as Graphics Engine using HepRep)
- Write User Documentation (Developper Documentation good using JavaDoc & Together)



HowTo Run

- **More stable but older version (from release):**

  # to get the Java environment:

  . $SRT_DIST/current/installed/share/scripts/Graphics_Java_env.sh

  # to start GraXML:

  $SRT_DIST/current/installed/$SRT_TARGET/bin/GraXML.sh [-o] [-i|-n] [-p|-q] [-b [vrmlview|vrml|xml|<external browser>]] <xml_file> [<xml_file>]

- **Recent version:**

  # to get the Java environment:

  . /afs/cern.ch/atlas/project/graphics/Java/setup.sh

  # to start GraXML:

  /afs/cern.ch/atlas/project/graphics/Java/GraXML.sh [-o] [-i|-n] [-p|-q] [-b [vrmlview|vrml|xml|<external browser>]] <xml_file> [<xml_file>]

http://atlas.web.cern.ch/Atlas/GROUPS/GRAPHICS/Texts/Documentation/GraXML/

JAS

(T.Johnson)

- New Release 2.2.1
- AIDA histograms can be transparently send from C++ to JAS ☒AIDA ☒AIDAC++
- HTL (AIDA) histograms from Objy can be accessed via JAS DIM
- AIDA => collaboration with Lizard and OpenScientis
- Soon Plugins:
  - BeanShell ☒BeanShell
  - Root DIM
- For Release 2.3:
  - N-tuple manipulation
  - 3D LegoPlot
  - AIDAML (AIDA XML)

JACO

(T.Johnson, M.Donszelmann)

- Java Access to C++ Objects
- C++ Parser + automatic glue Code generator ☒JACO
  - verified on MultiMethods from Lassi
  - verified on HTL
- Implementation for G4 exists ☒G4
- Implementation for Athena being prepared:
  - access to Event

Graphics in Atlas CVS/SRT

without WIRED, jAtlantis

- Java:
  - 150 classes
  - 34 kloc
  - (300kB bytecode)
- C++:
  - 148 classes
  - 30 kloc
- Fortran:
  - 78 files
  - 122 kloc

Atlantis

Java is part of the release (installed/share/java).

Graphics Domain is not yet represented in the new structure.

- bytecode includes src+classes+icons
- there are many dummy C++ classes, they are not needed in Java
- Java introduced into SRT via ad-hoc solution (not using SRT features)
- CERN Java Distribution is used