WIRED 4 - A GENERIC EVENT DISPLAY PLUGIN FOR JAS 3

M. Dönszelmann, Stanford Linear Accelerator Center, Stanford, U.S.A.

Abstract

WIRED 4 is an experiment independent event display plugin module for the JAS 3 (Java Analysis Studio) generic analysis framework. Both WIRED and JAS are written in Java.

WIRED, which uses HepRep (HEP Representables for Event Display) as its input format, supports viewing of events using either conventional 3D projections as well as specialized projections, such as a fish-eye or a ρ-Z projection. Projections allow the user to scale, rotate, position or change parameters on the plot as he wishes.

All interactions are handled as separate edits which can be undone and/or redone, so the user can try out things and get back to a previous situation. All edits are scriptable by any of the scripting languages supported by JAS, such as pnuts, jython or java itself. Hits and tracks can be picked to display physics information and cuts can be made on physics parameters to allow the user to filter the number of objects drawn into the plot. Multiple event display plots can be laid out on pages combined with histograms and other plots, available from JAS itself or from other plugin modules. Configuration information on the state of all plots can be saved and restored allowing the user to save his session, share it with others or later continue where he left off.

This version of WIRED is written to be easily extensible by the user/developer. Projections, representations, interaction handlers and edits are all services and new ones can be added by writing additional plugins. Both JAS 3 and WIRED 4 are built on top of the FreeHEP Java Libraries, which support a multitude of vector graphics output formats, such as PostScript, PDF, SVG, SWF and EMF, allowing document quality output of event display plots and histograms.

WIRED 4

WIRED 4 grew out of the ideas implemented in earlier versions of WIRED[1], together with the need of making it extensible by the user and easier to maintain. WIRED 4 features access to HepRep 1 and HepRep 2 data, user-defined projections, undoable edits/actions, direct and indirect interactivity, animated feedback on user's actions, an enhanced filtered picking mode, display of physics attributes, detector and event visibility selection and high quality vector graphics output. For some of these features WIRED 4 relies on either JAS 3 and or the FreeHEP Java Libraries. WIRED 4 is experiment independent, but can easily be extended and customized for an experiment. It is written in pure Java and runs on Windows, MacOsX, Linux and any other platform where the Java Runtime Environment has been ported.

JAS 3 PLUGIN MODULES

JAS 3 features a standard analysis tool[2], which can be extended by loading plugin modules. WIRED 4 is such a plugin module. It extends JAS 3 with an experiment independent event display. Experiment specific plugins may alter the behaviour of JAS and/or WIRED.

For much of the standard application features WIRED 4 relies on JAS 3, which in its turn relies on the FreeHEP Application Framework. Features of this framework include among others: the GUI, the menu system, the functionality to load and update plugin modules, file handling, event browsing and saving and restoring the state of the application.

DATA FORMAT

For its internal data representation WIRED 4 relies on the HepRep 2 standard[3]. HepRep 1 defines a data format for event display which consists of information such as tracks and hits and their visual representations as boxes, polygons, etc. Additional attributes can be attached to set drawing properties or specify physics information. The HepRep 2 standard adds hierarchical data types to better organize the HepRep information and it separates event data from detector geometry data to make event browsing easier.

HepRep data can be communicated via XML, CORBA or RMI. HepRep1 data can be converted to HepRep 2 using a backward compatibility plugin module. Other modules are available to read HepRep data from CORBA servers implementing the HepEventServer and or JProcman protocols.

Experiments can easily generate HepRep 2 data from their internal formats, by using one of the C++ or Java HepRep libraries to create in-memory HepReps and then handing them directly to WIRED 4 or writing them out to XML for later usage.

PROJECTIONS

WIREDs projection system allows the usage of conventional 3D projection, see figure 1, using matrix calculations as well as user-defined special projections[4][5]. These projections include but are not limited to: the parallel projection, which creates a real world picture of the detector and allows for scaling, rotation and translation, see figure 2 (top-left); the cylindrical fish-eye projection, which enlarges the center region of a cylindrical detector, while compressing the outer regions, see figure 2 (top-right); the ρ-Z projection, which cuts the detector along the z-axis and folds up the upper and lower halves, see figure 2 (bottom-left);

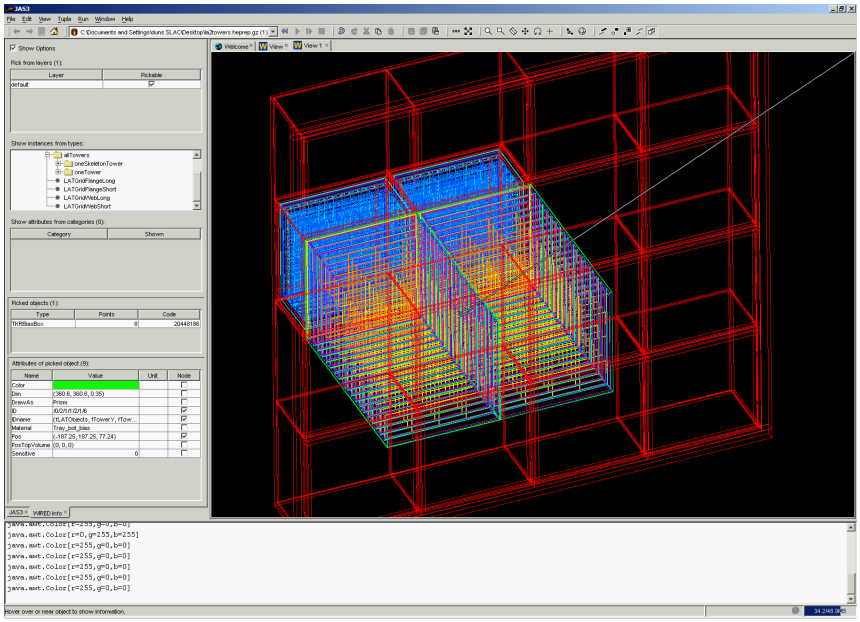

Figure 1: The GLAST detector, rotated using a 3D projection.

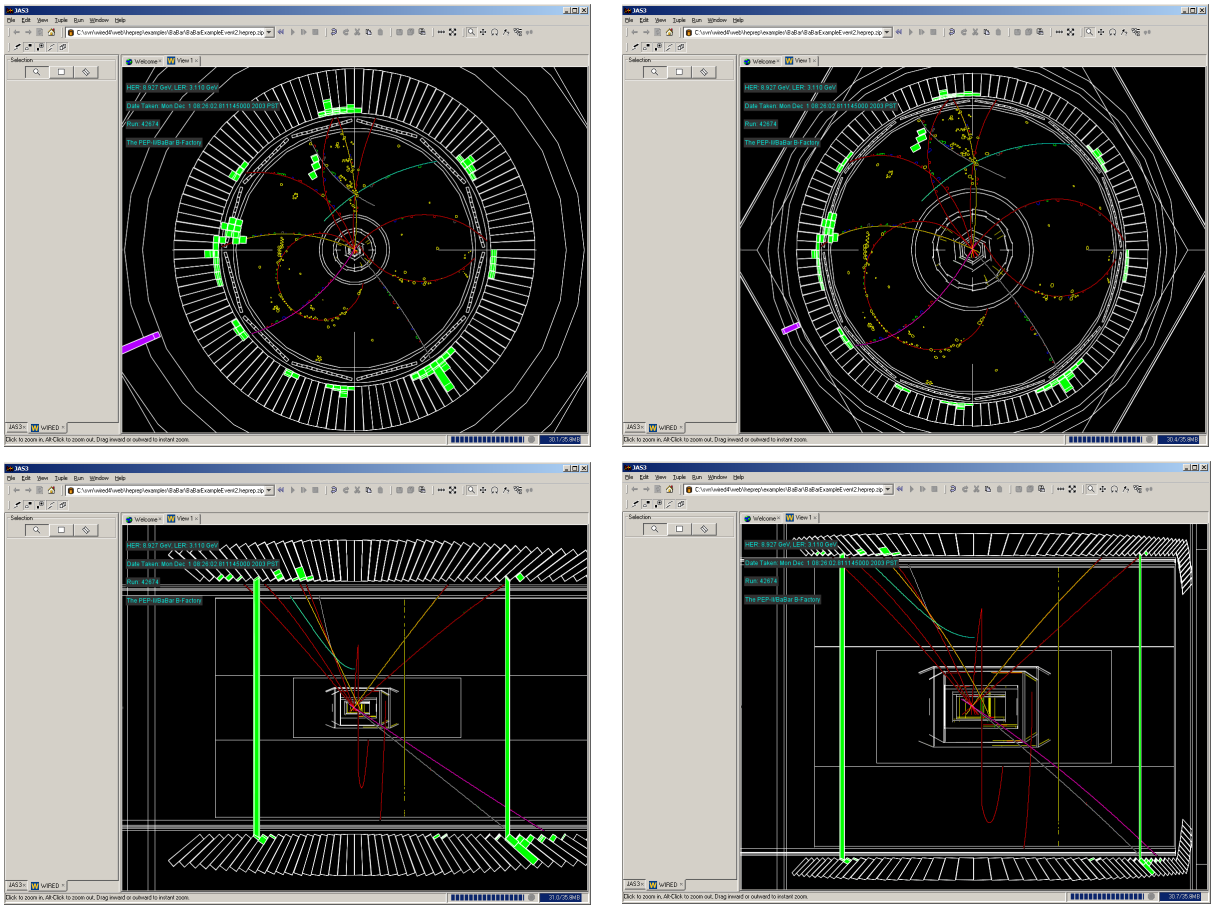

Figure 2: The BaBar detector and an event shown in a XY projection (top-left), shown in a fish eye projection (top-right), shown in a ρ-Z projection (bottom-left) and shown in a composite projection combining a fish-eye and a ρ-Z projection (bottom-right).

the variable projection which allows you to specify how to calculate u,v,w from x,y,z using any formula and number of variables and finally the composite projection, which combines any of the other projections by executing them in order, see figure 2 (bottom-right) for an example.

GRAPHICS ENGINE

To enable linear and non-linear projections, WIRED 4 uses its own graphics engine. The graphics engine draws straight into a graphics screen context from the HepRep 2 in-memory data, using a fast iterator which informs the graphics engine about any changes in drawing attributes. It uses a layered model to draw hits on top of tracks on top of calorimeter hits on top of geometry, making sure any object is always visible to the user.

The graphics engine is also used for drawing into special, non-visible, graphics contexts which calculate the bounding box, the nearest object to the cursor or the set of objects within an arbitrary shape. These contexts are used for picking, see below.

UNDOABLE EDITS

Changes to the plots in WIRED 4 are all made using edits, such as rotate, translate, but also the setting of a new projection. Most of these edits are reversible, meaning that the user can undo, and later on redo, the edit. This allows the user to experiment with the plots. Multiple edits of the same kind can be merged together.

INTERACTIVITY

The user can interact with the plots in ways that depend on the chosen projection(s). The parallel projection includes scaling, rotation and translation, but other projections may allow interactive setting of parameters.

Different interactivity modes are implemented by overlaying a transparent panel on top of the current plot. This panel handles all the mouse actions (click, drag, release, …) and is also used to draw shapes for animated feedback to the user. Using a separate panel for interactivity allows us not to interfere with the current plot thereby cutting down on the number of redraws.

WIREDs control panel to the side of the plots allows for more precise selection of the interactivity mode or to show output information.

PICKING

Picking of objects in WIRED 4 is supported as just another interactivity mode, and can be extended by the user. The standard picking mode allows you to pick either a single object or multiple objects within a rectangular shape. In either mode, the picked objects are shown in a table on the side. Selecting one of the picked objects will show physics and other attributes in a separate table below, see figure 3.

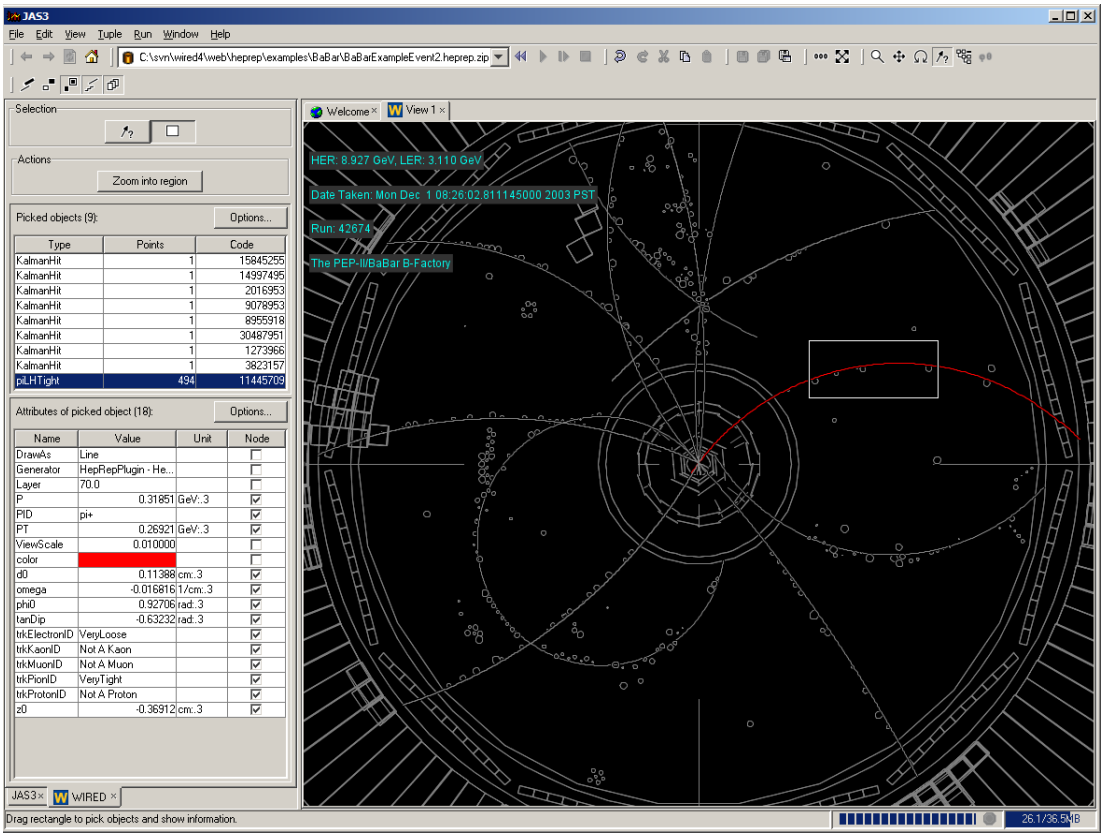

Figure 3: Picking multiple objects will display them in a table and show their physics attributes.

To show the user which objects are currently picked, the plot will draw all non-picked objects in grey, while leaving the picked objects in their original color. This way the picked objects look high-lighted.

Picking can be restricted by filters. Current filters pick objects from certain layers or of certain types. This allows you to pick only tracks and hits, while ignoring detector geometry. Attributes can also be filtered by type, to show for instance only physics quantities.

The picking mechanism is implemented by invisibly redrawing the current content of the plot into a special graphics context. This context calculates the closest object to the cursor, or the objects within a given shape. At the end of the drawing cycle, the list of objects is returned by the plot and put into the table.

SELECTION

The user may not be interested in displaying all the HepRep 2 data available for an event. WIRED 4 features a detector and event visibility tree showing both the HepRep 2 data types and individual data objects. The visibility of these types and objects can be toggled using this tree. The tree also allows you toggle the visibility of complete sub trees if you wish.

OUTPUT

The FreeHEP Vector Graphics libraries[6] support a set of extensible Export Formats. These libraries are used by WIRED 4 to support both copy-paste and save-as behaviour.

On Windows platforms copy-paste uses EMF (Enhanced Meta File) while on MacOsX it uses PDF (Portable Document Format), both of which are vector graphics formats. On Unix platforms copy-paste falls back to one of the supported bitmap formats. Pasting of these formats can be done using for instance any of the standard Office products.

WIRED 4 allows you to save the graphics produced in the following vector graphics formats: PostScript, PDF, EMF, SVG (Scalable Vector Graphics), SWF (Macromedia Flash) and CGM (Computer Graphics Metafile). Common bitmap formats are also supported, such as GIF, PNG, JPG and BMP.

WIRED 4 can also save the HepRep 2 data of a plot into an XML file for later usage.

EXTENDING WIRED 4

WIRED 4 can be extended by the user in many ways. You can write your own edit, projections, representations, interaction handlers and export formats. You can also write your own implementation for the plot and the graphics engine if you wish.

All extensible items are defined as service interfaces and WIRED 4 implements them as such. If you implement a service, wrap your classes up as a separate plugin and install it in JAS 3, WIRED 4 will automatically find these extensions and add these services to the correct place (menu, list of representations, list of projections, …), next to where WIRED 4 keeps its own services.

Another way of getting more out of WIRED 4 is by writing scripts in any of the JAS 3 supported scripting languages: Python, Pnuts or Java.

CONCLUSIONS AND FUTURE

WIRED 4 is a complete rewrite of earlier versions of WIRED to make the system better maintainable. The WIRED 4 system works as a plugin module in JAS 3 and inherits a number of features from it. HepRep 2 is used as the internal data format, but compatibility modules are available to access HepRep 1 data. WIRED 4s current features include standard projections, user interactivity, an enhanced picking mode and high quality output. All these features can very easily be extended by the user.

In the near future WIRED 4 will be extended by a general scripting capability compatible with JAS 3, interactive filters and cuts, object labelling, and scales.

ACKNOWLEDGEMENTS

I would like to thanks the other members of the JAS 3 and FreeHEP team, Tony Johnson, Victor Serbo and Max Turri, for their help and support in letting WIRED 4 get the maximum functionality out of JAS 3. Also many thanks to all the students and others who worked on earlier versions of WIRED. Special thanks go to Hans Drevermann for his ideas on non conventional projections and to Paolo Palazzi for initiating the WIRED project and suggesting its name.

REFERENCES

[1] WIRED, WWW Interactive Remote Event Display, http://wired.freehep.org

[2] JAS, Java Analysis Studio, http://jas.freehep.org

[3] HepRep, High Energy Physics Representables for Event Display, http://heprep.freehep.org

[4] H.Drevermann, D.Kuhn and B.S.Nilsson, “Is there a Future for Event Display?”, Proc. of the 1992 CERN School of Computing, L’Aquila, Italy, CERN Yellow Report 93-03.

[5] J.Drohan et al., “The Atlantis Visualization Program for the ATLAS Experiment”, Proc. of Computing in High Energy Physics, 2004.

[6] The FreeHEP Java Libraries, http://java.freehep.org