WIRED
World-Wide Web Interactive Remote Event Display

M.C. Coperchio [a], M. Dönszelmann [b], N. de Groot [c],
P. Gunnarsson [d], M. Litmaath [b], D. McNally [b], N. Smirnov [e]

[a] *Dipartimento di Física, Università di Bologna and INFN, Via Irnerio 46, I-40126 Bologna, Italy*

[b] *CERN, CH-1211 Geneva 23, Switzerland*

[c] *SLAC, P.O.Box 4349, Stanford CA 94309, California, USA*

[d] *Fysikum, Stockholm University, Box 6730, S-113 85 Stockholm, Sweden*

[e] *Inst. of High Energy Physics, Serpukov P.O.Box 35, 142284 Protvino, Russia*

The WIRED project: http://www.cern.ch/WIRED

WIRED (World-Wide Web Interactive Remote Event Display) is a framework, written in the Java™ language, for building High Energy Physics event displays. An event display based on the WIRED framework enables users of a HEP collaboration to visualise and analyse events remotely using ordinary WWW browsers, on any type of machine. In addition, event displays using WIRED may provide the general public with access to the research of high energy physics.
The recent introduction of the object-oriented Java™ language enables the transfer of machine independent code across the Internet, to be safely executed by a Java enhanced WWW browser. We have employed this technology to create a remote event display in WWW. The combined Java-WWW technology hence assures a world wide availability of such an event display, an always up-to-date program and a platform independent implementation, which is easy to use and to install.

Key words: HEP Event Displays; WWW; Java; CORBA.

1 Introduction

Today's High Energy Physics experiments are typically carried out by big collaborations of many institutes and universities spread over several countries.

Each of these institutes is equipped with a variety of computers to run analysis jobs, interactive tools and event displays. All institutes are connected to the Internet, giving physicists access to the events of their experiment. To enable physicists to look at these events from any machine anywhere in the world, WIRED[1,2] was created as an event visualisation framework using Java™[3] inside a World-Wide Web browser[4], as shown in figure 1. This work was initially performed in view of the DELPHI[5] detector at CERN, and later generalised for other HEP experiments.

WIRED can be accessed world wide. By clicking on a link any authorised user may download a full event display and interactively browse events. Because WIRED is written in Java, it is portable across all platforms that provide Java compatible WWW browsers. There is no need for any installation (apart from the WWW browser), since parts of the WIRED program will be transferred automatically when needed. Whenever a user downloads the event display, he gets the most up-to-date version of the code. WIRED is also optimised for

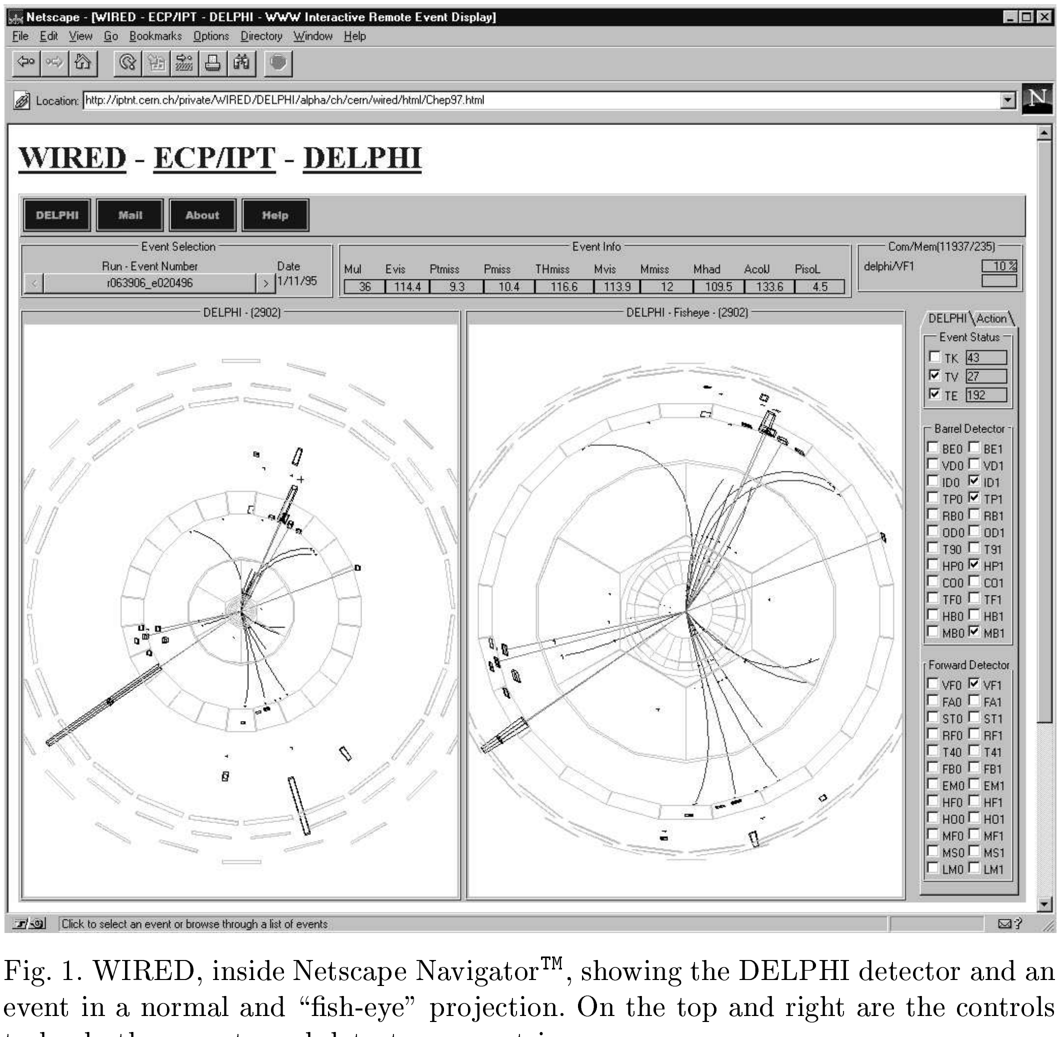

Fig. 1. WIRED, inside Netscape Navigator™, showing the DELPHI detector and an event in a normal and “fish-eye” projection. On the top and right are the controls to load other events and detector geometries.

network use: most of the code is downloaded before the user starts viewing events. Data are then loaded as the need arises. Refer to table 1 for an indication of the size of the code, the data and the time it takes to load. Local resources, such as CPU, memory and graphics accelerators, are then used for interactive operations like zooming.

Table 1
Code and data sizes, their approximate loading times and server types.

	Size	Download time (over slow network)	Server Type
Code (classes)	$\sim$ 200 kb	$<$ 2 minutes	HTTP
Event	$<$ 50 kb	$<$ 25 seconds	HTTP
	$>$ 50 kb	$<$ 5 s/part (loaded in parts)	CORBA
Detector Geometry	10–50 kb per subdetector	5–25 seconds	HTTP

Currently WIRED provides the following functions. It is possible to view multiple different projections of the same or different events. This allows for visual comparisons as well as studies at different levels of detail. Parts of the event and detector geometry may be selected and will be loaded on demand. They can be rotated and zoomed. Context-sensitive help is available inside the browser. The user interface is configurable to allow for full-fledged technical as well as easy to use educational configurations.

2 Architecture

The WIRED architecture is designed to minimise network traffic, to use local resources where possible, to be configurable for different user needs and to accommodate different detectors. WIRED provides an adaptable object-oriented framework and allows for a designer to use default implementations or to construct his own. For example, the DELPHI detector may have its own data structure and therefore its own data conversion and visualisation modules. WIRED consists of both a server and a client. The server runs continuously on a computer that has access to the data, while the client is in fact an Applet ("small" application), running inside the browser, as shown in figure 2.

The server provides the Applet with event and detector geometry data. These data may be distributed over a number of servers in different formats. Furthermore, multiple data access mechanisms are used. Sequential access to the geometry data may be provided by an ordinary HTTP server. For events, which may vary significantly in size (refer to table 1), CORBA[6] is used to transfer only those pieces of data in which the user is interested, thereby keeping the network load to a minimum. The use of CORBA allows for the server to be written in a different language than Java. Legacy data access packages can thus be interfaced with WIRED.

The Applet uses local resources to allow the user to interact with the event. It consists of a store to cache the data, a set of *viewing pipes* to filter and convert the data, and the graphical user interface (GUI) to display them.

The store caches the data in a format depending on the experiment. The store may use different loaders to load data in multiple formats, or it may use CORBA to pick up specific objects and cache them in the Applet. Data may be stored directly as tracks and hits, or may be converted immediately into drawable parts. The end situation is always the same: the client side has a (reduced) copy of the server data available in local memory. This copy is used by the viewing pipes to convert the data and make them visible in the GUI.

The viewing pipe is divided into two parts. First the event and geometry data are converted into 3D polyhedrons. Different conversions are used for hits, tracks and energy depositions in calorimeters, and experiment extensions are possible. Selections of hits, tracks and energy depositions take place in this part of the viewing pipe.

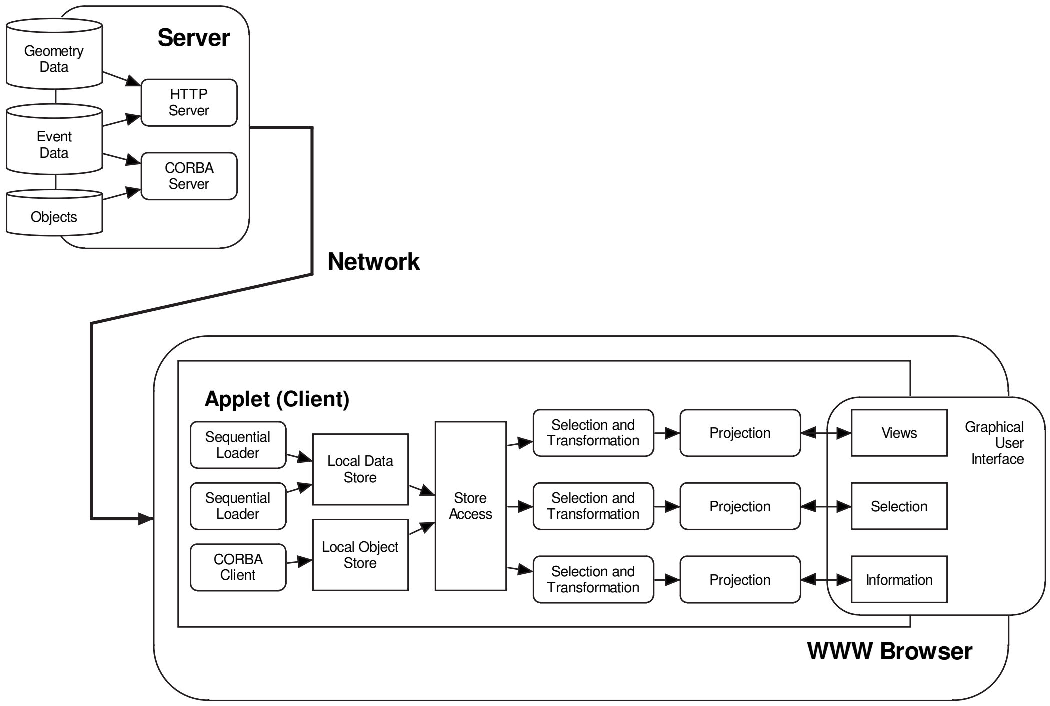

Fig. 2. The architecture of WIRED: the server, the applet and the flow of information.

The 3D polyhedrons are then projected onto a 2D plane to make them drawable on the user's screen. Different projections[7] from 3D to 2D are foreseen, such as parallel, perspective and fish-eye projections. Rotations, zooming and translations take place in the projection part of the viewing pipe, and are implemented using conventional matrix calculations[8]. Apart from the 2D projection, an event can also be visualised as a list of tracks and hits. The same viewing pipe is used here, except that the selection and projection are more straightforward.

The user interface of WIRED consists of a set of components: the view, the detector selector, the event selector, etc. Each of these components is configurable by a set of parameters, such as the initial projection, event number, etc. The components are positioned by WIRED like lego blocks onto the screen, and into the GUI hierarchy, as shown in figure 3. This layout is described by a small configuration file, which is read when WIRED is loaded. New configurations can be loaded at any time. The technology used here is further described in DUI[9].

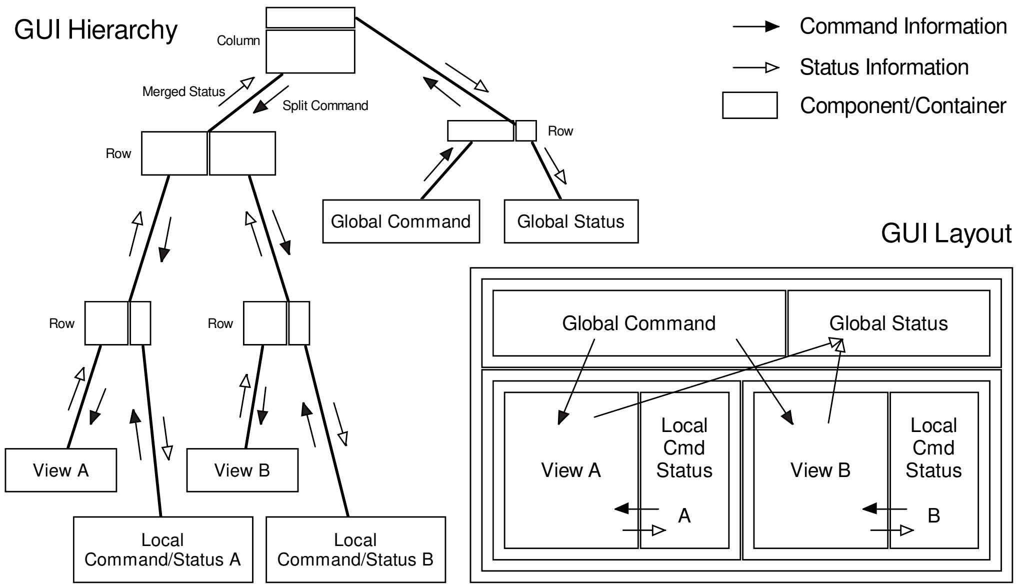

Fig. 3. An example configuration of WIRED: the GUI and its components, its hierarchy and the information channels between the components.

The information channels needed for event selection, geometry selection, rotations and zooming are set up by connecting the GUI components with each other and with the store. WIRED uses a publish-subscribe mechanism for the connections and Java's introspection to find out where each of the components has to be connected. The connections are made based on a policy given by the GUI layout, thereby logically grouping components with their views. Components can have a command and status character. Global command components send their commands to all views, while local command components only act on the nearest view. Status information flows from its view into the nearest as well as the global status components.

In this architecture CORBA provides for random access to different data formats, arriving from different servers. The use of Java provides for a secure downloadable Applet, which may dynamically instantiate a variety of loaders. The multi-threading features of Java[10] are extensively used to push data from the network, via the store, through the viewing pipe into the GUI and onto the screen. This progressive viewing gives people the idea of faster access to their data. Java's introspection together with a configuration file parser generated by the JavaCUP[11] and JavaLex[12] packages allow for a fully configurable graphical user interface.

3 Status and future developments

In its initial configuration WIRED was used for the DELPHI experiment (august 1996). It has evolved since into a more generic framework, which is available to a number of experiments. In particular the L3[13] and Chorus[14] experiments (both at CERN) and the BaBar[15] experiment (at SLAC) are currently participating in the WIRED project and their setup is well underway. Use of the future versions of the Java Development Kit[16] is foreseen, as well as an enlargement of the functionality of WIRED and a tighter cooperation with the data and reconstruction packages of the experiments.

4 Conclusion

WIRED is a portable, easily configurable and extendible framework to write event displays for HEP experiments. The choice of Java as a language has solved most of the porting and distribution problems of older systems. The object-oriented nature of Java makes extending the framework for different experiments a fairly easy task. With the inclusion of the CORBA technology to access and interact with the data WIRED will gain in analysis power.

References

[1] M.C. Coperchio, M. Dönszelmann, P. Gunnarsson, F.L. Navarria, T. Rovelli, *WIRED, A WWW Interactive Remote Event Display proposal*, **BOLOGNA-DFUB/96-9** (1996).

[2] M. Dönszelmann, M.C. Coperchio, P. Gunnarsson, WIRED - World-Wide Web Interactive Remote Event Display: A Status Report, *Proceedings of the HEPVIS'96 workshop* (Geneva, Switzerland, 1996).

[3] J. Gosling, B. Joy, and G. Steele, *The Java™ Language Specification* (Addison-Wesley, 1996).

[4] T.J. Berners-Lee, R. Cailliau, J.F. Groff, and B. Pollermann, World-Wide Web: The Information Universe, *Electronic Networking: Research, Applications and Policy* **2**(1) (1992) 52–58.

[5] P. Aarnio et al. (Delphi Collaboration), The Delphi Detector at LEP, *Nucl. Instr. and Methods in Physics Research* **A303** (1991) 233-276.

[6] The Object Management Group and X/Open, *Common Object Request Broker: Architecture and Specification* (John Wiley & Sons, Inc., 1994).

[7] H. Drevermann, D. Kuhn, B.S. Nilsson, *Event Display: Can we see what we want to see?*, **CERN-ECP/95-25** (1995).

[8] J.D. Fowley et al., *Computer Graphics: principles and practice, 2nd Edition* (Addison-Wesley, 1996).

[9] M. Dönszelmann, C. Gaspar, and J.A. Valls, A Configurable Motif Interface for the DELPHI Experiment at LEP, *Proceedings of the International Motif User Conference '92* (Washington D.C., USA, 1992) 156–162.

[10] D. Lea, *Concurrent Programming in Java™: Design Principles and Patterns* (Addison-Wesley, 1997).

[11] S. Hudson, Java based Constructor for Useful Parsers (CUP), *http://www.cc.gatech.edu/gvu/people/Faculty/hudson/java_cup/home.html* (Georgia Institute of Technology, Atlanta, USA, 1996).

[12] E.J. Berk, Java-Lex: A lexical analyzer generator for Java, *http://www.cs.princeton.edu/~appel/modern/java/JavaLex/* (Princeton University, Princeton, USA, 1996).

[13] L3 Collaboration, B. Adeva et al., *Nucl. Instr. and Methods in Physics Research* **A289** 35 (1990).

[14] M. de Jong et al. (Chorus Collaboration), *A new search for $\nu_\mu - \nu_\tau$ oscillation*, **CERN-PPE/93-131** (1993).

[15] The Babar Collaboration, *BaBar Technical Design Report*, **SLAC-R-95-457** (1995).

[16] The Java Development Kit, *http://www.javasoft.com/products/JDK/* (Sun Microsystems, 1997).