JClarens: A Java Based Interactive Physics Analysis Environment for Data Intensive Applications

Arshad Ali[1], Ashiq Anjum[1], Tahir Azim[1], Michael Thomas[2], Conrad Steenberg[2], Harvey Newman[2], Julian Bunn[2], Rizwan Haider[1], Waqas ur Rehman[1]
[1]National University of Sciences and Technology, Rawalpindi, Pakistan
{arshad.ali, ashiq.anjum, tahir.azim, rizwan.haider, waqas.rehman}@mail.niit.edu.pk
[2]California Institute of Technology, Pasadena, CA 91125, USA
{thomas ,conrad,newman}@hep.caltech.edu, Julian.Bunn@Caltech.edu

Abstract

In this paper we describe JClarens; a Java based implementation of the Clarens remote data server. JClarens provides web services for an interactive analysis environment to dynamically access and analyze the tremendous amount of data scattered across various locations. Additionally this research is aimed to develop a service oriented Grid Enabled Portal (GEP) that provides interface and access to several Grid services to give a homogeneous and optimized view of the distributed and heterogeneous environment. Other than showing platform independent behavior provided by Java, the use of XML-RPC based Web Services enabled JClarens to be a language neutral server and demonstrated interoperability with its Python variant. Extreme care has been taken in the usage and manipulation of various Java libraries to cater the needs of high performance computing. The overall exercise has yielded in a prototype with strong emphasis on security and virtual organization management (VOM). This shall provide a common platform to support development of larger, more flexible framework with future aims to integrate it with a loosely coupled, decentralized, and autonomous framework for Grid enabled Analysis Environment (GAE).

1. Introduction

In today's world, computing has become increasingly collaborative and multidisciplinary. It is not unusual for teams to span institutions, states, countries, and continents. The web provides very basic information sharing mechanisms that help such groups to work together. But what if they could link their data, computers, sensors and other resources into a single virtual machine? The emerging Grid technologies seek to make this possible. In the past, distributed computing in particular was considered as a way to share basic computing resources. Grid computing, most simply stated, is distributed computing taken to the next evolutionary level. The goal is to create an illusion of a simple yet large and powerful self-managing virtual computer out of a large collection of connected heterogeneous systems sharing various combinations of resources. Pioneered in an e-science context, Grid technologies are also generating interest in industry, because of their apparent relevance to commercial distributed-computing applications [1].

Grid computing has already begun to play a role in scientific applications. Scientific computations require reliable transfer of data in distributed heterogeneous environments. These can consist of parallel programs sending large, complex, and rapidly changing data objects or self-contained modules sending events to steer other modules. Scientific systems also have complex run-time systems designed for heterogeneous environments with dynamically varying loads and multiple communication protocols. Java is an ideal technology well suited for a role in the success of such computational systems, as it is capable of handling high-performance messaging and leveraging the benefits of high-speed networks. The development of the Java-based JClarens web services framework aims to use these capabilities of Java to create an interactive analysis environment for data intensive applications.

1.1. Grid Analysis Environment

The initial uses of the Grid have been in areas of batch production processing and simulation. An additional area of Grid work, interactive analysis, is now under way. Current analysis tools require the user to run tasks on a single machine on data accessible to that machine. The dream of interactive analysis on the Grid is that the user, with the push of a button, might move a job from a single node to a distributed environment, access more power and more data, and do it all seamlessly, so that his work is no harder on the Grid than it was on a single machine. Figure 1 describes the visual flow of the services within one such analysis environment being developed at Caltech: the Grid Analysis Environment (GAE) [2].

The GAE focuses on the construction of such infrastructure that allows scientists to interactively perform analysis and steer the execution of various jobs and tasks during the analysis process. Under the hood, it is not as easy to run interactive analysis on the Grid as it is on a single machine. The same data may be replicated in many locations, competition for resources is much more complex, the number of higher-priority tasks than ones own is not automatically known, and therefore the best choice of how and where to execute a task is hard to determine. For these reasons, new forms of Grid services that are able to make reasonable choices among a range of possible job-execution strategies, autonomously or interactively, are needed. Decisions made by these services will be based on a more complete range of information about the Grid's current and future state, and may integrate user-Grid information exchanges as part of the decision process.

The Java based JClarens is part of the larger GAE architecture and provides a portal interface to a general collection of Grid services, along with a more specific collection of anlaysis services. Furthermore, this infrastructure offers its services to a variety of clients that include the traditonal desktop GUI, command line tools, all the way down to clients running on resource-limited handheld devices.

JClarens is being developed as a Java-based version of the Clarens Grid-enabled web services framework. Clarens is designed to provide a framework that allows new Web services to be registered and deployed with ease in a Wide Area Network (WAN). It aims to provide powerful Virtual Organization (VO) management, while maintaining architectural simplicity. Clarens is envisioned to act as the "backbone" within the GAE, and will host the VO and lookup services. In addition, Clarens interfaces can be developed easily for various Grid components to allow them to act as web services in a WAN, using the authentication and VO capabilities of Clarens. In this way, Clarens can provide wrappers for various Grid components to act as Web services, and provide interoperability between these components.

JClarens is being developed keeping the same objectives and design principles in mind. A modular, object-oriented design has been developed for JClarens, which allows new services to be added with ease. The VO, authentication and lookup services of JClarens are described in Section 2. JClarens and the original Python based Clarens will act as complementary, interoperable Grid service hosts. Depending on the platform on which a particular Grid component is based, an interface for it can be provided either with the Python or the Java based Clarens, allowing it to act as an interoperable Web service over a wide area network. JClarens, in particular, will act as a very suitable and convenient platform for hosting Java-based Grid software (such as Sphinx and MonALISA) as Web services.

Thus, unlike most other particle physics projects, JClarens does not focus on developing Grid-enabled physics applications or services. Instead, JClarens is meant to act as a server capable of hosting all these services in a Grid environment, and exposing their functionality through simplified programmatic interfaces. This enables the development of simpler, lightweight clients capable of carrying out complex, interactive analysis activities on the Grid.

In this paper, we present a Java based infrastructure for High Performance computing to facilitate the remote analysis of data generated by Compact Muon Solenoid Dectector (CMS) [3]. This enables physicists at the European Organization for Nuclear Research (CERN) [4] and users at remote sites to execute jobs, manipulate data and files, and use components of the computational Grid through a Web interface.

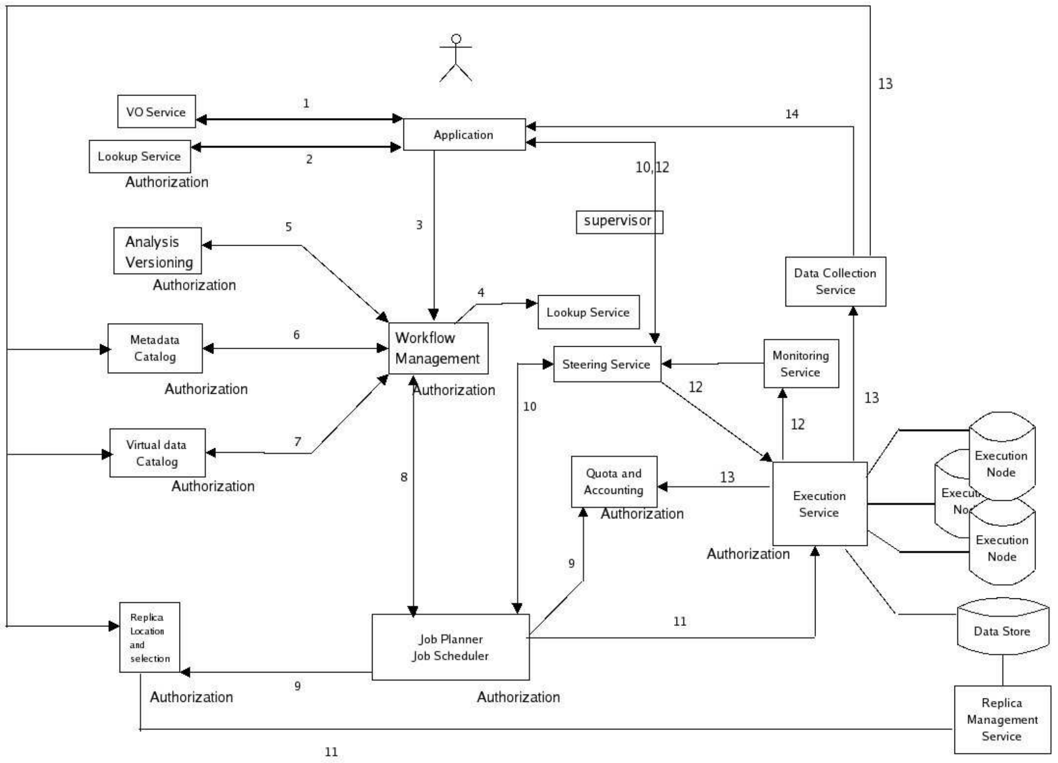

Figure 1. Interaction of different services in GAE

2. Architecture

As stated earlier, the Java based JClarens acts as a web service portal, a single access point to various Grid services, network resources and other scientific applications. As a result, JClarens is implemented following a layered architecture in order to achieve greater scalability and save development time. Open source tools have been used to make it robust and widely acceptable. Technologies like Apache Axis [5], Apache JetSpeed [6], Apache Tomcat [7], Grid Security Infrastructure (GSI) [8], and MySQL [9] are used in developing its architecture, which is depicted in Figure 2. Grid computing has been merged into portal computing to fulfill some specific requirements. Its portal behavior hides the complexities of Grid technologies from the user and presents simplified, intuitive interfaces for harnessing the power of the underlying resources. This architecture helps in focusing on maintaining the services rather than allocating resources to users in a complex way. Apache JetSpeed was used in JClarens to give portal behavior and graphical interface to this application. JetSpeed was also integrated with Tomcat to provide the desired functionality.

The following subsections are devoted to the description of various features and services that have been developed as part of JClarens.

2.1. Security

Security holds a supreme importance when resources or services are made publicly available. In addition, the Grid is a place that is concerned with sharing and coordination of diverse kind of resources in distributed "virtual organizations" [10]. Moreover, the user will be accessing heterogeneous resources; in that case it will be cumbersome to supply a password again and again before using different resources.

Therefore, single sign-on along with security has been desired in order to allow the user to authenticate once, irrespective of the number of resources one needs to access. Hence, the challenge of building a secure Java based JClarens is to define an architecture that allows the integration of such security mechanism without compromising security and integrity of other computational resources. To match the desired requirements and extensively address these security issues, Grid Security Infrastructure (GSI) was adopted as a perfect candidate. The service enables easy coordination of multiple resources, authenticating users once and letting them perform multiple actions without re-authentication. Once authenticated by GSI, the client is able to access other resources or services.

2.2. Authentication

The authentication procedure is initiated by invoking the RPC method system.auth() with username and password as part of the HTTP Basic Authentication header. The server responds with a list of:

I. Its certificate,
II. The server session ID encrypted using the user's public key, and
III. The client session ID encrypted using the server's private key.

This ensures that only some one in possession of the client's private key can discover the server session ID, and also that the server is in possession of a private key matching the certificate sent as (I) above. Once the certificate and session ID exchange is complete, both the client and server certificates can be verified against the publicly available CA certificate chain, verifying that each is who they claim they are.

2.3. Authorization

The system module implements fine-grained access control of all methods that are available on the server through a set of access control lists (ACLs). This is done by organizing users, uniquely identified by their distinguished names (DNs), into a hierarchical virtual organization (VOs) of groups and subgroups.

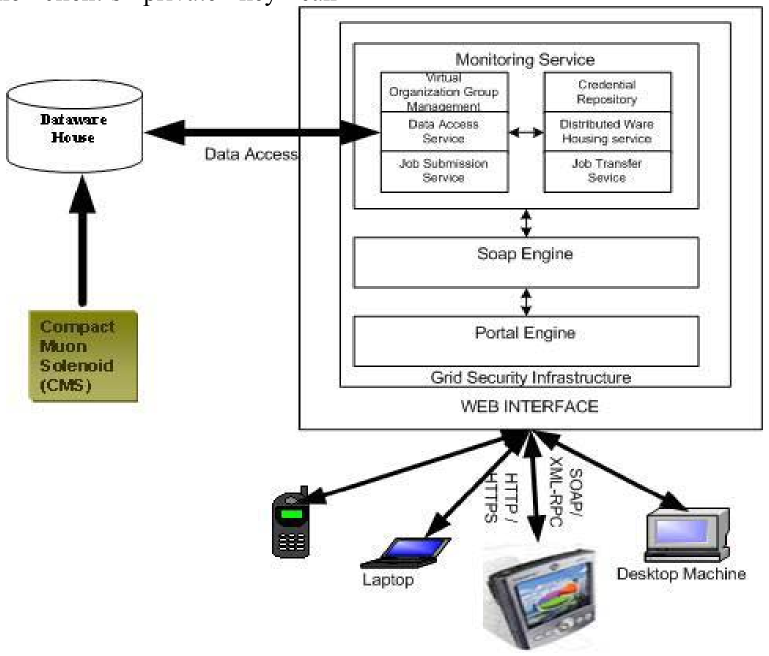

Figure 2. Architecture Diagram of JClarens

2.4. Services

A set of services that has been developed for JClarens for carrying out tasks such as Virtual Organization (VO) management, job submission, proxy credential storage, file transfer and service lookup etc. is described below:

2.4.1. Virtual Organization Group Management. VO Group Management is a mechanism provided by JClarens facilitating the distributed authorization management while maintaining individual preservation of individual identity and Grid identity, established by the user's home institute/organization. Groups are defined based on certificates issued by a Certifying Authority (CA) as shown in Figure 3. Later, at a Grid site, these groups are mapped to users on the local system via a gridmap file, which is similar to ACL. A detailed description of VO management in Clarens and JClarens is given in [11]

2.4.2. Credential Repository. A Credential Repository service, similar to MyProxy, has been developed to serve this purpose. MyProxy has itself not been used in order to remove dependencies on Globus and other components, that MyProxy requires. A MySQL based repository has been established to manage the credentials.

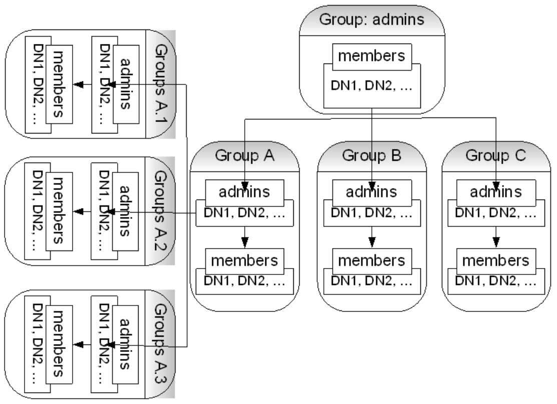

Figure 3. Virtual Organization Architecture

2.4.3. File Access. The file access service provides programmatic access to files in a similar manner as FTP but provides access to files controlled via a set of access control lists. Methods are provided to browse the file system, download files, get file sizes, search within files, get modification information, and obtain md5 hash values of files to ensure integrity.

2.4.4. Monitoring Service. Grid-based data analysis requires information and coordination among services and computational resources. There will be differences in the computational, storage and memory capabilities of computers across the Grid. A monitoring service will therefore be required to provide information about the best available resources for job execution. Additionally, the monitoring service will be used to provide information regarding the status of executing jobs. Currently, a monitoring service for JClarens based on MonALISA [12] is in the development stages.

2.4.5. Lookup Service. A lookup service has been implemented that allows Clarens servers to look up services from other Clarens servers. This ensures that one server going down at any time does not cause the whole system to crash. It also makes sure that certain crucial services (available in all Clarens servers) are always up.

Currently the lookup service is centralized, which means that if the central registry database goes down, then the lookup service will also fail. However, a decentralized peer-to-peer architecture is now being implemented, which will ensure a greater degree of fault tolerance.

3. Interoperability

One of the biggest challenges was to develop a Java based server that can also interoperate with the existing python-based Clarens implementation. This was required so that the Python and Java implementations can offer a complementary set of services (Sphinx scheduling in JClarens, POOL persistency [13] in PClarens), and servers written in the two languages can then interoperate with each other. This was also required so that clients can communicate with both the implementations without changes in their source code.

Java facilitated us in accomplishing this requirement since it does not depend on different platforms or operating systems. Moreover, a Java based XML-RPC library offers an opportunity to extend the foundation of network-centric development environment in a structured way. The new infrastructure was implemented successfully providing the required interoperability with the existing Python based server implementation.

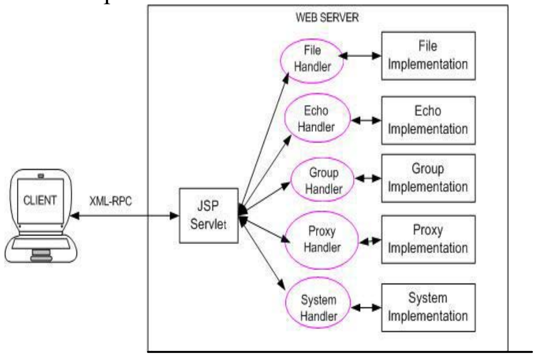

Figure 4: JClarens Interoperability

4. Comparisons and Evaluations

Given the wide variety of problems to be solved, finding the right language that solves the widest possible variety of programming chores was a task of paramount importance.

Our research has shown that for this particular type of application, Java provides maximum efficiency and programming support. A shorter development time and powerful Web programming features make Java ideally suited for programming this type of software. Additionally Java tools tend to appear first in certain areas of active research, such as XML and Web services.

We have performed various tests to check the efficiency of the newly developed JClarens. All the times are an average of 3 executions. Table 1 shows the result between Python and Java. These tests all include the startup time for the JVM. If we focus on the time spent running the benchmark, we can deduct about a second from the Java scores (and almost nothing for Python). This is particularly relevant for the 'no', 'speed' and 'native' benchmarks.

Table 1. Time Comparison between Java and Python

	Python(s)	Java 1.4 (s)
Console	22.93	33.58
Hash	34.84	6.35
Io	33.16	3.68
List	31.05	2.71
No	0.12	0.86
Speed	31.81	1.18
Native	33.97	1.4

4.1 Echo Performance

Echo test is a simple mechanism of testing the availability of the server. It accepts any basic type (int, double, string) and returns what we provide in the input. The call made was through a Java client. It first called a Java implementation of the server (JClarens), after which it called the Python-based implementation (PClarens). In each case, five hundred XML-RPC calls to the server were made.

Table 2. Echo Performance

Test Name	JClarens	PClarens
TestEchoPerformance	3.1	7.927

Table 2 clearly shows that the Java based implementation of the Clarens server was more efficient than the Python based implementation, as JClarens took 3.1 s to respond to 500 calls, whereas PClarens took almost 8 s.

Several other test were performed in order to measure the efficiency between the Python based PClarens and Java based JClarens. Table 4, Figure 5, Table 5 and Figure 6 show the output of various tests.

From the outcome, it is clear that the Java based version works faster for most of the tests. However, in certain important tests it is slower as compared to the Python version. In particular, the file transfer method (file.read) in PClarens is several times faster than in JClarens. Since file transfer is one of the most important features offered by Clarens, some work definitely has to be done to improve the performance of this method.

Table 3. Evaluation of Various Methods

Method Name	JClarens (sec)	Pclarens (sec)
echo.echo	0.03	0.037
group.add_admins	0.184	0.192
group.add_users	0.127	0.209
group.admins	0.164	0.203
group.users	0.137	0.192
group.delete	0.129	0.181
group.delete_users	0.13	0.19

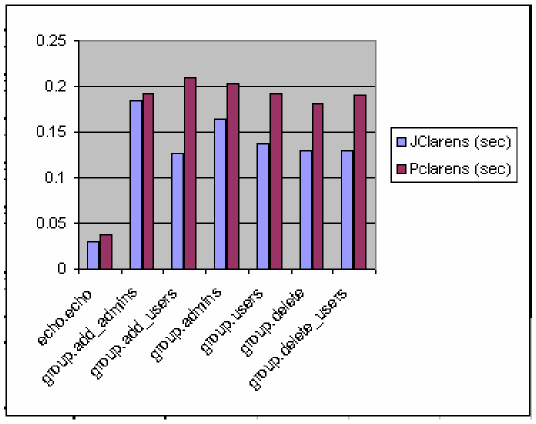

Figure 5. Performance Graph of Methods

Table 4. Time taken by Additional Test

Method name	JClarens (sec)	Pclarens (sec)
proxy.list	0.198	0.214
proxy.list_admin	0.224	0.218
proxy.store	0.194	0.2
proxy.delete_admin	0.208	0.224
proxy.delete	0.192	0.578
proxy.retrieve	0.191	0.249
group.create	0.189	0.154
group.list	0.187	0.175
system.add_acl_deny	0.113	0.193
system.set_acl_specs	0.115	0.171
system.get_acl_specs	0.125	0.162
system.get_acl_names	0.03	0.154
system.del_acl_specs	0.104	0.173
system.list_methods	0.032	0.133
system.method_signture	0.036	0.144

system.method_info	0.033	0.143
system.auth	0.122	0.708
file.read (1 MB file) - 5 iterations	0.602	0.192

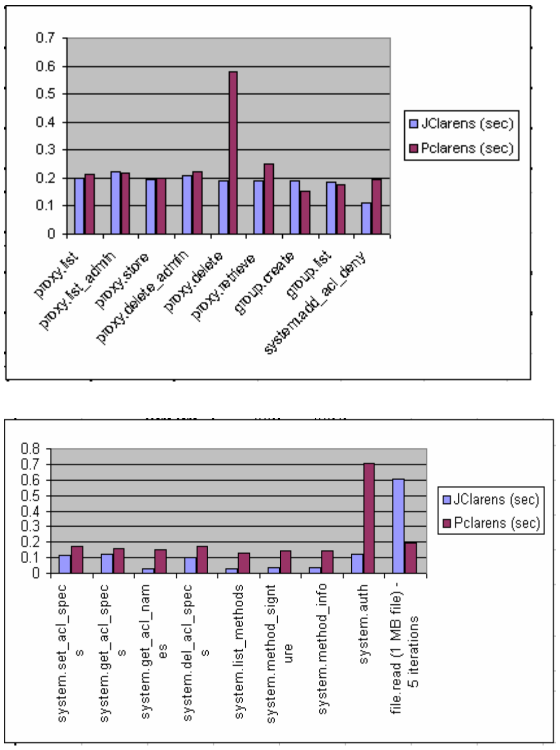

Figure 6. Graph of various additional Performance Tests

5. JClarens Clients

The idea of JClarens server was to provide an easy and robust interface for various services to a variety of clients. A wide range of clients in Python, Java, C/C++ etc have been developed, used, and tested.

5.1. JAS

Java Analysis Studio (JAS) [14] is developed at Stanford Linear Accelerator (SLAC). JAS aims at carrying out the analysis of high-energy physics data and allows the user to perform arbitrarily complex data analysis tasks by writing analysis modules in Java.

A JClarens client for JAS has been developed. Authentication of the client is carried out using the system.auth method, while data searching, browsing, and downloading services are provided by the 'file' service.

5.2. JASOnPDA

JASOnPDA [15] is the scaled down version of Java Analysis Studio, especially designed for constrained handheld devices. JASOnPDA provides essential analysis utilities of Java Analysis Studio on PocketPC devices, and was developed using J2SE 1.1. As it has been built for mobile users, JASOnPDA has the additional facility to log on to JClarens server using a certificate-based authentication procedure.

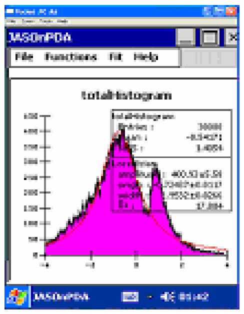

Figure 7. JASOnPDA running on a PDA showing its features of histogram plotting, function fitting, and statistics

Once successfully authenticated, the user is allowed to access files stored at the server. This client uses Clarens to remotely browse and download ROOT files. As shown in figure 7, it then analyzes them to draw various analysis histograms.

5.3. WWW Interactive Remote Event Display (WIRED)

WIRED [16] is one of the first Event Displays written in Java for use on the World-Wide-Web. It provides a framework for writing event displays. A prototype plug-in has been developed for WIRED that enables it to authenticate with JClarens, search for and download HEPREP files, and render the event and detector geometry stored in those files.

5.4. WIREDONPDA

WiredOnPDA is another analysis application developed for PocketPC devices. WiredOnPDA accesses data using JClarens in the same way as JASOnPDA. Once authenticated, the user can access JClarens and select any HepRep2 physics event data file placed on the JClarens server. Figure 8 displays WIRED running on PDA.

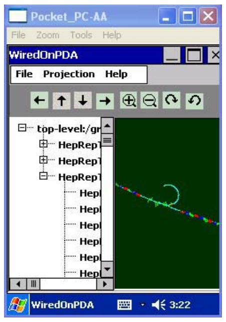

Figure 8. WiredOnPDA displaying event data and the structure of a HepRep2 file in separate panes.

6. Future Developments

Work is underway to extend the functionality and provide enhanced and better features like OGSA Compliance etc. JClarens is also being integrated with heterogeneous and distributed data warehouses using multi-agent technology for efficient information retrieval from the repositories. An automatic resource planning and reservation system for the GAE is also under research to be plugged into JClarens. Moreover, it is being optimized to process and carry very huge datasets in the range of many Gigabytes per second.

7. Conclusion

The future for Clarens (both Java- and Python-based) [17] is very promising as Grids in general and High Performance computing in particular will soon become commonplace for small and large communities of scientists linking their various resources to support human communication, data access, and computation.

JClarens offers a simple architecture providing many of the basic services needed to construct Grid applications such as security, resource discovery, resource management, UDDI based lookup and data access. It holds a growing set of services and useful client implementations. Using JClarens, the capability to expose various software components as Grid services will become much easier. This makes JClarens an extremely important component of the Grid Analysis Environment (GAE), and a step towards the attainment of a more interactive Grid environment.

8. References

[1] Foster, I., Kesselman, C., Nick, J. and Tuecke, S. The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration, Globus Project, 2002. http://www.globus.org/research/papers/ogsa.pdf
[2] Grid Analysis Environment web site (http://ultralight.caltech.edu/gaeweb)
[3] The Compact Moun Solenoid Home (CMS) Site http://cmsinfo.cern.ch/Welcome.html
[4] CERN http://public.web.cern.ch/public/
[5] The Large Hadron Collider Home Page http://lhc-new-homepage.web.cern.ch/
[6] Apache AXIS http://ws.apache.org/axis/
[7] Apache Jetspeed http:// jakarta.apache.org/jetspeed/
[8] Apache Tomcat http:// jakarta.apache.org/tomcat/
[9] The MySQL Site http://www.mysql.com
[10] Foster, I., Kesselman, C., Nick, J. and Tuecke, S. The Anatomy of Grid: Enabling Scalable Virtual Organizations. *International Journal of High Performance Computing Applications, 15 (3).200-222.2001.*
[11] *Conrad D. Steenberg, Eric Aslakson, Julian J. Bunn, Harvey B. Newman, Michael Thomas, Frank van Lingen.* The Clarens Web Services Architecture. Proceedings of CHEP 2003, paper MONT008, 2003.
[12] MonALISA (http://monalisa.cacr.caltech.edu)
[13] D.Düllmann, M. Frank, G. Govi, I. Papadopoulos, S. Roiser. The POOL Data Storage, Cache and Conversion Mechanism. *Computing in High Energy and Nuclear Physics 2003, San Diego, March 24- 28, 2003*
[14] Java Analysis Studio (http://jas.freehep.org/)
[15] Ashiq A., Ali A., Azim,T. Investigating the Role of Handheld devices in the accomplishment of Interactive Grid-Enabled Analysis Environment. Grid and Cooperative Conference, Shanghai, 2003.
[16] WWW Interactive Remote Event Display (http://wired.freehep.org/)
[17] Clarens web pages (http:// www.clarens.sourceforge.net)