
HPCC Year One Progress Report

"A Distributed, Real-time Hurricane Wind Analysis System"

Hurricane Research Division NOAA/AOML, Miami, FL

The following is a progress report for the first year of a three year project entitled "A Distributed, Real-time Hurricane Wind Analysis System". The first year of this project has been funded by NOAA's High Performance Computing and Communications Program, with Matching funds provided through the FEMA-NIBS HAZUS Project.

Updated* Milestones and Deliverables

1. **06-01-98**

Completion of rapid prototype application for dynamically querying the HURDAT hurricane storm database and delivering query results and storm track plots to the web. The web client application accessible from most modern Web browsers will be considered as the deliverable.

2. **06-15-98**

Complete operational build of the 1998 HRD Wind Analysis System.

3. **07-01-98**

Complete updates to scripts to download and process observations in real time.

4. **09-01-98**

Complete feasibility testing of distributed object technology. Complete testing of IDL for automatic product generation.

5. **10-01-98**

Report on first year project progress.

6. **07-01-98**

Completion of Object-Relational Database evaluation

* The project commenced when the HPCC funds were recieved in February 1998.

Progress Report

1. Completion of rapid prototype application for dynamically querying the HURDAT hurricane storm database and delivering query results and storm track plots to the web. The web client application accessible from most modern Web browsers will be considered as the deliverable.

Documentation:

We have developed a rapid prototype application demonstrating dynamic and platform independent

delivery of information from an Oracle 8 database with object extensions to the World Wide Web using the JAVA language. Both the database and the web components of the prototype application are hosted by HRD workstations situated outside of AOML's firewall (<http://storm.aoml.noaa.gov/Storms/index.html>). The client applet allows for dynamically querying the HURDAT ("Best Track") hurricane storm database and delivering query results and storm track plots to the web and should be accessible from most JAVA 1.1.x capable PC or workstation platforms with internet access.

Unfortunately some sites may be temporarily unable to connect for a number of reasons. If a client is running from inside of a firewall, the applet may not be able to access the AOML database without an Oracle SQL*Net proxy in place on their firewall. Also, in some instances, the browser may not be running a version of the Java Runtime Environment (JRE) required to run the applet correctly. For these instances, we have installed a Java Plugin on the web server for updating client browsers to a compatible JRE. The plugin is freely available from JavaSoft and currently only works on Windows95/NT and Solaris based Netscape 3, Netscape 4 and Internet Explorer 4 browsers. Other platforms will follow. In addition to the suggestions made above, users connecting to our applets are also encouraged to download the latest version of a Java enabled browser for up to date JRE and plugin compliance and have at least 32 megabytes of RAM on their machines. The applet performs correctly when it is run outside of the browser using the appletviewer application from a user installed 1.1.x JRE.

HRD Best Track Storms Page [Screen Grab](#)

We have completed this milestone ahead of the projected June 1st deadline.

2. Complete operational builds of the workstation-based QC Client and distributed analysis system software for use during the 1998 hurricane season. Scientists at HRD will use the operational builds to provide realtime products to NHC while development efforts move forward throughout the year.

Documentation:

Development for both the QC Client and Analysis Automation subsystems is well under way. In general, all code is up to date with latest jdk release (1.2 beta3) and implementation of most basic functionality is complete with the notable exceptions being in the area of database integration. We are currently using a test database for extracting and manipulating test data, but committing edited and new data to that database has not yet implemented. Also, the triggering of an analysis from a QC client has yet to be implemented, but should not be a difficult task.

Specifically, most of the client application subproject functionality requirements have been met. Some of these requirements include: 1) map loading and drawing, 2) plotting of wind observations and storm track fixes in both synoptic and storm relative coordinates, 3) graphical tools such as zooming, observation flagging (quality control), distance calculation and detailed observation inspection and editing, 4) separate "views" panel for immediate graphical response to any changes to data.

We have developed a generic set of Java packages (groups of compiled classes) to promote code reuse among all Java implementations by our development team. Code reuse is important for testing, minimizing errors, saving space and saving time. The current upper level packages include "hrd.geography" (classes representing geographical objects such as global positions, global areas, maps, etc...), "hrd.meteorology" (classes representing meteorological objects such as wind observations, storm track fixes, storm tracks, etc...), "hrd.math" (classes representing mathematical objects such as vectors, matrices and angles), "hrd.apps" (classes and subpackages pertaining to applications and the various subsystems of the project), and the utility packages "hrd.util" (non-graphical utilities) and "hrd.awt.util" (graphical utilities).

The documentation of a Java development effort is made easier through self documenting source code. The javadoc program available in the JDK creates HTML API references based on developer source code comments with one command. A current API reference, class hierarchy and index of methods and attributes is available to developers at all times.

HRD Wind Analysis Client [Screen Grab](#)

This task is on track to meet the projected June 15th deadline, but may be slightly late if other tasks require more attention.

3. Update scripts for downloading and decoding real-time wind observations into the HRD wind analysis system from remote locations at NHC, NCEP, and other sites. Update and enhance processing algorithms to ensure that all input data conform to a common framework for height, averaging time, and exposure. Add new aircraft- and space-based remote sensing wind observing platforms as they become available.

Documentation:

We have completed work on decoders for land-based METAR data, and sea based C-MAN, Bouy, and Ships data that we will be receiving from NHC's family of services feed. These decoders are written in standard ANSI C for portability and performance reasons. Aircraft and satellite data decoders have not yet been coded in C, but are next on our agenda. In addition to decoders, we have also ported various proven legacy boundary layer conversion routines from FORTRAN to C. These routines and decoders deliver data in formats appropriate for ingestion by our database and by HRD's legacy file-based analysis system.

This task is on track to meet the projected July 1st deadline.

4. Perform feasibility testing of distributed object technology. Encapsulate the FORTRAN components of the legacy objective wind analysis algorithms in the C language on a remote analysis server (Solaris or HPUX workstation), and then have a prototype JAVA client and application server communicate with this server and the remote database server.

Documentation:

An Objective Analysis run consists of 4 steps equivalent to 4 major FORTRAN programs that run both on HP-UX 10.x and Sun Solaris 2.x platforms. In order to make each of these steps callable from any operating system/architecture on the Internet, they have to be distributed as objects. We accomplish this by wrapping (linking together and calling one program with another) the FORTRAN code in JAVA, an object oriented programming language. This development has, so far, been done exclusively on AOML machines. At the time that we were wrapping the FORTRAN code, only the Solaris 2.x platform had a version of the operating system that allowed the installation of the proper tools, mainly the latest version of the Java Development Kit. Therefore, after finding out that the version of HP-UX that AOML had available (HPUX 9) was not sufficient, we concentrated our efforts on prototyping distributed objects on Solaris.

Using Java Native Interface (JNI) and C, we were able to control the call (with and without parameters) and return of the FORTRAN programs to the main Java application. Then, by using JacORB, a freeware CORBA 2.0-compliant Object Request Broker (ORB), we created the interface that advertises the objects. To test these two crucial steps, we ran the all the Analysis steps with the client machine located in Taiwan and the analysis server located onsite at AOML (Miami, FL). Specifically, we ran a Java program located on a Sparc machine in Taiwan that invoked a request to the server in Miami. The server then executed the three main subprograms required to complete an analysis. Each subprogram is a separately wrapped unit of code. The analysis ran quite well and helped show us that a CORBA distribution scheme will indeed serve the project well. CORBA allows applications to communicate with one another no matter where they are located or who has designed them. Using an ORB, the middleware that establishes the client-server relationships between objects, a client can transparently invoke a method or function on a server object, which can be on the same machine or across a network. We are also investigating Java's Remote Method Invocation (RMI) technology for use in a heterogenous (Java client to Java server) distribution scheme.

The HRD owned machine designated as the Analysis server at NHC has recently been upgraded to HP-UX 10.20 and now contains the requisite FORTRAN and C compilers to allow us to replicate the AOML Solaris implementation. Because we are using Java, a cross-platform portable language, it should not take very long to replicate the analysis system on both platforms.

This task is on track to meet the projected September 1st deadline if it is not met ahead of schedule.

5. Begin initial design of product suite and development of routines for product delivery. These routines take the HRD Wind analysis results and deliver georeferenced images representing various fields such as wind swaths. Web access to these routines and GIS conformity among products is a priority. Investigate Interactive Data Language (IDL), Common Gateway Interface (CGI), and a generalized Java interface for dynamic product delivery.

Documentation:

Work on product generation routines will be completed in concert with the demand for new products. We have completed development on the subset of routines that represents the products that HRD had available in realtime in previous years. All the routines are now implemented in Research Systems' (RSI) Interactive Data Language (IDL). The routines are capable of plotting images to a local screen and/or generating images in a variety of file formats for delivery as hard copies. We are investigating different methods of delivering dynamically generated IDL products over the web. Both CGI and Java based methods are possible, but software licensing, product delivery speed and the intricacy of the code are still issues to be resolved for each proposed method.

This task is on track to meet the projected September 1st deadline.

6. Complete evaluation of off-the-shelf relational, object-relational, and object-oriented database management systems. Procure and begin training for implementing and testing the chosen Database Management System with conventional atmospheric observations.

Documentation:

The fundamental tasks of designing, feeding, storing, extracting, and managing data in a database requires an analysis on the data requirements. A database is a data repository that provides a centralized and homogeneous view of data for multiple applications. HRD conducted a data analysis for the Hurricane Wind Analysis System and designed a logical schema, a global view of the database. The system's schema is specified in terms of the semantic data modeling. A semantic data model is a mechanism for specifying the structure of the database and operations that can be performed on the data in the database. From our analysis, we concluded that a modern database management system was mandatory to serve Web applications to end-users. HRD has been conducting independent evaluations of available "off the shelf" relational, object-relational, and object-oriented commercial DBMSs including Oracle8, OpenBase, Mini SQL, NETCDF, OSMOS, Informix Universal Server.

We have, furthermore, developed a database application prototype, the stormtrack database (<http://storm.aoml.noaa.gov/Storms/index.html>) and have elected to use an Oracle8 database server with object extensions to implement it. Oracle8 was chosen based on advantages in manageability, performance, scalability, transaction management and fault tolerance. We purchased Oracle8 Workgroup for WindowsNT (5-users) and database training units to train our application developers and database administrator on Oracle8.

This milestone should be met before its projected July 1st deadline.

7. Budget

The project is currently on budget. Matching funding from the National Institute for Building Science will arrive after AOML and ERL finance work out procedures. A potential long term issue is the number

of concurrent database user licenses required for efficient use of the analysis system. We will explore this issue further by testing application performance under the current 5 user license.

Funded by NOAA [High Performance Computing and Communications \(HPCC\)](#)