NOAA Technical Memorandum NOS CS 7

Description of *nowCOAST* Version 2, a GIS Web Mapping Portal to Real-Time Coastal Observations and NOAA Forecasts

Silver Spring, Maryland
May 2007

noaa National Oceanic and Atmospheric Administration

U.S. DEPARTMENT OF COMMERCE
National Ocean Service
Coast Survey Development Laboratory

Office of Coast Survey
National Ocean Service
National Oceanic and Atmospheric Administration
U.S. Department of Commerce

The Office of Coast Survey (OCS) is the Nation's only official chartmaker. As the oldest United States scientific organization, dating from 1807, this office has a long history. Today it promotes safe navigation by managing the National Oceanic and Atmospheric Administration's (NOAA) nautical chart and oceanographic data collection and information programs.

There are four components of OCS:

The Coast Survey Development Laboratory develops new and efficient techniques to accomplish Coast Survey missions and to produce new and improved products and services for the maritime community and other coastal users.

The Marine Chart Division acquires marine navigational data to construct and maintain nautical charts, Coast Pilots, and related marine products for the United States.

The Hydrographic Surveys Division directs programs for ship and shore-based hydrographic survey units and conducts general hydrographic survey operations.

The Navigational Services Division is the focal point for Coast Survey customer service activities, concentrating predominately on charting issues, fast-response hydrographic surveys, and Coast Pilot updates.

NOAA Technical Memorandum NOS CS 7

Description of *nowCOAST* Version 2, a GIS Web Mapping Portal to Real-Time Coastal Observations and NOAA Forecasts

Micah Wengren
Michael Allard
Matt Kennedy
John G. W. Kelley
Meredith Westington

May 2007

noaa **National Oceanic and Atmospheric Administration**

U. S. DEPARTMENT OF COMMERCE
Carlos Gutierrez, Secretary

National Oceanic and Atmospheric Administration
Conrad C. Lautenbacher, Jr., VADM USN (Ret.), Under Secretary

National Ocean Service
John H. Dunnigan
Assistant Administrator

Office of Coast Survey
Captain Steven R. Barnum, NOAA

Coast Survey Development Laboratory
Mary Erickson

NOTICE

Mention of a commercial company or product does not constitute an endorsement by NOAA. Use for publicity or advertising purposes of information from this publication concerning proprietary products or the tests of such products is not authorized.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS

.axl	Map Configuration File
.dbf	dBASE File Containing Attribute Information of Features
.shp	Feature Geometry File
.shx	Hashtable of .shp/.dbf files
4DWX	UCAR's 4-Dimensional Weather Observing Network
AJP	Apache JServ Protocol
AHPS	Advanced Hydrological Prediction Service
AIRMAP	Atmospheric Investigations, Regional Modeling, Analysis and Prediction
ArcIMS	Arc Internet Map Server

ArcSDE	Arc Spatial Data Engine
ArcXML	Arc eXtensible Markup Language
ASOS	Automated Surface Observing System
AWOS	Automated Weather Observing System
BLP	Boundary Layer Profiler
BML	Bodega Marine Laboratory
C-MAN	Coastal-Marine Automated Network
Caro-COOPS	Carolinas Coastal Ocean Observing and Prediction System
CBOS	Chesapeake Bay Observing System
CIO	Chief Information Officer
CMN	Coastal Monitoring Network
COMPS	Coastal Observing and Prediction System
CO-OPS	Center for Operational Oceanographic Products and Services
CORIE	COlumbia RIver Estuary
CREWS	Coastal Reef Early Warning System
CRN	Climate Reference Network
CSDL	Coast Survey Development Laboratory
CSS	Cascading Style Sheets
DBF	See .dbf
DBI	Database Interface
DEP	Department of Environmental Protection
DHTML	Dynamic Hyper Text Transfer Protocol
DNS	Domain Name System
DOD	Department of Defense
EEZ	Exclusive Economic Zone
EMPACT	Environmental Monitoring for Public Access and Community Tracking
EPA	Environmental Protection Agency
ESDIM	Environmental Services Data and Information Management
ESRI	Environmental Systems Research Institute
ETSS	Extra-Tropical Storm Surge Model
FGDC	Federal Geographic Data Committee
FME	Feature Manipulation Engine
FTP	File Transfer Protocol
GIS	Geographic Information System
GLERL	Great Lakes Environmental Research Laboratory
GoMOOS	Gulf of Maine Observing System
GSOS	GPS Surface Observing System
HF	High Frequency
HPCC	High Performance Computing and Communications
HTTP	Hyper Text Transfer Protocol
ICON	Innovative Coastal-Ocean Observing Network
IFLOWS	Integrated Flood Observing and Warning System
IOOS	Integrated Ocean Observing System
JAVA VM/JVM	Java Virtual Machine
JDK	Java Software Development Kit

JRE	Java Run Time Environment
JSP	Java Server Pages
LAIS	Louisiana Agriclimatic Information System
LEO	Long-term Ecosystem Observatory
LIShore	Long Island Shore Network
LUMCON	Louisiana Universities Marine CONsortium
MDDNR	Maryland's Dept. of Natural Resources water quality monitoring station network
MMAB	Marine Modeling and Analysis Branch
MMAP	Marine Modeling and Analysis Programs
MDL	Meteorological Development Laboratory
MDOT	Maryland Department of Transportation
MLML	Moss Landing Marine Lab
MOS	Model Output Statistics
MS	Microsoft, Inc.
MVCO	Martha's Vineyard Coastal Observatory
NCEP	National Centers for Environmental Prediction
NCCOOS	North Carolina Coastal Ocean Observing System
NDBC	National Data Buoy Center
NDFD	National Digital Forecast Database
NEOCO	Network for Environmental Observations of Coastal Ocean
NERON	NOAA's Environmental Real-Time Observation
NEXRAD	NEXt Generation RADar
NIFC	National Interagency Fire Center
NMS	National Marine Sanctuaries
NOAA	National Oceanic and Atmospheric Administration
NOS	National Ocean Service
NWIS	National Water Information System
NWLON	National Water Level Observation Network
NWS	National Weather Service
OCS	Office of Coast Survey
ODAAS	Operational Data Acquisition and Archive System
ODBC	Open Data Base Connectivity
OOS	Office of Operational Systems
OPC	Ocean Prediction Center
PNG	Portable Network Graphics
PORTS	Physical Oceanographic Real-Time System
PWSN	Prince William Sound Observing Network
RAOBS	Upper-Air Atmospheric Soundings (radiosondes)
RAM	Random Access Memory
RAWS	Remote Automated Weather Stations
RFC	River Forecast Center
RTMON	Real-time Meteorological Observation Network
QPF	Quantitative Precipitation Forecast
SABSOON	South Atlantic Bight Synoptic Offshore Observational Network

SEAKEYS	N/A
SHEF	Standard Hydrometeorological Exchange Format
SIT	Stevens' Institute of Technology
SPO	Special Projects Office
SSI	Server-Side Includes
SSL	Secure Sockets Layer
SQL	Structured Query Language
TABS	Texas Automated Buoy Station
TCOON	Texas Coastal Observation Network
TPC	Tropical Prediction Center
UCAR	University Corporation for Atmospheric Research
UrBANet	N/A
URL	Uniform Resource Locator
USF	University of South Florida
USGS	United States Geological Survey
VMC	Vermont Monitoring Cooperative
WADOT	Washington Department of Transportation
WAVCIS	Wave-Current-Surge Information System
WFO	Weather Forecast Office
WSR	Weather Service Radar
WWW	World Wide Web
XML	Extensible Markup Language

ABSTRACT

The Marine Modeling and Analysis Programs (MMAP) branch of NOAA/National Ocean Service's (NOS) Coast Survey Development Laboratory (CSDL) has developed a Geographic Information System (GIS)-based Web mapping portal called *nowCOAST* (http://nowcoast.noaa.gov). This portal allows both MMAP model developers and the nation's coastal community as a whole to quickly find and access online, real-time observations and NOAA forecasts for any region in the coastal United States. The portal provides users one-stop access to real-time meteorological, oceanographic, river, and water quality observations as well as NOAA forecasts for seaports, estuaries, the Great Lakes, and the coastal ocean.

nowCOAST provides access to this information through its spatially enabled database containing hyperlinks to thousands of pre-existing Web pages that display observations or NOAA forecast products at specific locations. By aggregating this information in a centralized database and displaying it in a dynamic user-controlled map via a Web browser, *nowCOAST* provides a new means for discovery of on-line, real-time information for coastal areas throughout the U.S. Access to this information is important for model development, coastal monitoring and forecasting, hazard assessment and response as well as for commercial shipping, recreational boating and other activities in the coastal zone. *nowCOAST* was first made available to the public in August 2002.

This report describes *nowCOAST* Version 2 which was released on January 3, 2005. Version 3 was implemented on November 9, 2005 and featured on-map display of satellite cloud imagery, surface meteorological and oceanographic observations, and weather radar reflectivity. *nowCOAST* Version 3 will be documented in a future NOAA/NOS technical report.

1. INTRODUCTION

1.1. Purpose of *nowCOAST*

An estuary or coastal region is affected daily by changes in weather, oceanographic, and river conditions. NOAA and other federal, state, and educational observing networks monitor the physical changes within many estuaries in real-time. NOAA numerical prediction models forecast future short-term changes in weather, oceanographic, and river conditions. In addition, NOS is developing the capability to forecast oceanographic conditions in selected estuaries. In order to develop these forecast models, real-time meteorological, oceanographic, and river observations are required. Identifying online sources of such real-time data for a particular geographic area can be a difficult and time-consuming process since each observing organization maintains a separate website and there is no single online source displaying geographic locations of real-time observing network stations.

In order to expedite the process of finding of real-time data for model development, MMAP built a mapping Web portal called *nowCOAST* based upon Geographic Information Systems (GIS) software Arc Internet Map Server (ArcIMS). *nowCOAST* uses a GIS database to store geographic positions of observing network stations and forecast locations, the type of observations taken or forecasts produced at each site, and links to Uniform Resource Locators (URLs) to pre-existing Web pages that display this information. The *nowCOAST* map provides controls for users to quickly specify the type of observations or forecasts they want as well as their geographic area of interest. From the resulting map users can view their selected data by linking directly to an external website.

nowCOAST's database includes information on real-time meteorological, oceanographic, river, and water quality observations from federal, state, and regional land and ocean observing networks. The portal also has links to NOS astronomical tidal predictions for coastal locations nationwide, point guidance from NOAA meteorological, estuarine, ocean, and river forecast models, as well as National Weather Service (NWS) weather forecasts, marine weather forecasts for coastal, offshore, and international waters, and also surf zone forecasts for selected U.S. coastal areas.

Although originally designed to assist MMAP model developers, *nowCOAST* was made available to the entire coastal community. The first version of *nowCOAST* was made publicly available on the Web in August 2002. The second version which is described in this report was released on January 3, 2005.

1.2. Application of GIS Internet Mapping Technology

The Web mapping portal was constructed using a commercial off-the-shelf GIS software package named ArcIMS along with several software languages such as Hypertext Markup Language (HTML), JavaScript, Java, and Perl. The ArcIMS software allowed the portal to be designed to serve two general classes of users, experienced and novice GIS users. For the inexperienced user, the *nowCOAST* 'Jumpbar' may be utilized to obtain real-time observations or forecasts. The Jumpbar consists of four Web-standard 'pulldown' menus (Fig. 2). These allow the user to specify

1) location, e.g. seaport, coastal state, estuary, lake, marine sanctuary,
2) information, i.e. specified type of observation, forecast guidance, or forecast,
3) variable, e.g. water level, air temperature, wave height, etc.,
4) time, i.e. specific forecast time or time period.

The user can then click on the observation or forecast site on the map to display the information. For the more experienced GIS user, the full suite of ArcIMS functionality is available, allowing for direct navigation of map extent and position, dynamic GIS layer-overlay and direct interaction with the data by query, buffer, select and find functions. This full functionality enables *nowCOAST* to provide users with powerful GIS tools to locate real-time data and forecasts in ways previously not possible on the Web.

1.3. Components of *nowCOAST*

ArcIMS provides the dynamic mapping functionality for *nowCOAST*. There are two major ArcIMS-based components customized for *nowCOAST*: the MapService and the map viewer application. The MapService is a Web service published through ArcIMS that is controlled by a specification file that defines the appearance of *nowCOAST's* data on a map. The *nowCOAST* map viewer is a Web application that is downloaded to the user's Web browser and provides the functionality necessary to interact with the MapService.

The non-ArcIMS software components of *nowCOAST* include software needed to run ArcIMS on the *nowCOAST* server as well as a collection of scripts and other programs that are used to perform, at regular intervals, updates to *nowCOAST* content at regular time intervals. The underlying Web server software required for ArcIMS includes Java, the Apache HTTP Web server, and the Jakarta Tomcat Servlet engine.

A detailed description of the ArcIMS and non-ArcIMS components is provided in the next section.

2. *nowCOAST* SERVER: DESCRIPTION OF ARCIMS AND NON-ARCIMS COMPONENTS

2.1. Hardware Specifications

NowCOAST runs on a high-end Dell Precision workstation with a 2.2Ghz Intel Xeon processor with 1 gigabyte (GB) of random access memory (RAM) and 10 GB of storage. Windows was selected as the operating system due to the availability of Windows support within NOS' Office of Coast Survey (OCS). This configuration does not provide for future scalability, but is adequate for the Internet traffic that *nowCOAST* currently receives.

2.2. ArcIMS Software Components

ArcIMS consists of several components that are installed on the ArcIMS server in order for the ArcIMS Mapservice to properly serve spatial data over the Internet. The Application Server and the Spatial Server are basic components of ArcIMS and must be installed in all ArcIMS server setups. The Java Connector is an optional piece of software that is used in *nowCOAST*. Presently, *nowCOAST* uses ArcIMS version 4.0.1.

2.2.1 Application Server

The ArcIMS Application Server is responsible for managing ArcIMS on the server. It runs as a background process on the server and handles load distribution of incoming requests, catalogs which services are running on which ArcIMS Spatial Servers, and routes Arc eXtensible Markup Language (ArcXML) requests to particular Spatial Servers.

The Application Server can only handle requests passed from the web server in ArcXML. This means that a request from the client viewer must either be in ArcXML or it must be interpreted and translated into ArcXML by a connector application before it reaches the Application Server. There are four connectors for ArcIMS, each specific to a particular type of viewer. *nowCOAST* uses the Java Connector, which is an add-on module to the Jakarta-Tomcat Servlet engine that translates Java ServerPages (JSP) and Java code into ArcXML. More information about the Java Connector is in Section 2.2.3.

2.2.2. Spatial Server

The ArcIMS Spatial Server is considered the "backbone" of ArcIMS since it processes a user's requests for maps and related information. The Spatial Server consists of a variety of components that together encompass the functionality of ArcIMS to serve maps and spatial data across the Internet to the user. These components are all labeled 'servers' themselves and consist of the following: Image Server, Feature Server, Query Server, Extract Server, Geocode Server, Metadata Server, and ArcMap Server. The number and type of each server implemented in the Spatial Server depends on the requirements of an ArcIMS site. *NowCOAST* currently relies only on an Image Server to produce maps for the client and a Query Server to handle data queries from the client.

2.2.3. Java Connector

The ArcIMS Java Connector is a set of JavaBeans that are executed within the Jakarta-Tomcat Servlet engine. For simplicity, the Jakarta-Tomcat Servlet engine will be referred to as Tomcat for the rest of the document. The Java Connector is typically installed as part of a Java web application that is served over the Internet via Tomcat. The Java Connector allows the web application to communicate with the ArcIMS Application Server. It translates requests from the Java web application into ArcXML, which is then interpreted and responded to by the ArcIMS Application Server. The Java Connector provides a custom JSP tag library to allow for development of JSP web applications.

More information about the Java Connector is available at:
http://support.esri.com/index.cfm?fa=knowledgebase.documentation.listDocs&PID=16

2.3. Non-ArcIMS Software Components

2.3.1. Web Server

The Apache HTTP (Hypertext Transfer Protocol) server was selected to run *nowCOAST*. Apache is a free, open-source web server that is the most commonly used on the Internet. It provides easy customization as well as superior security, and integrates well with Tomcat, which is discussed in the next section. *nowCOAST* Ver. 2 uses Apache version 2.0.47.

Information about the Apache HTTP server is available at: http://www.apache.org/

2.3.2. Servlet Engine

Some software components of ArcIMS require a servlet engine to run. Servlet engines are Java-based web servers that function as a bridge between the web server (Apache) and any Java code that is running on the web server. Commonly, Java code that runs on the server consists of Java Servlets or JSPs and provides the ability to serve dynamic web content. Servlets are a popular way to create platform-independent web-based applications.

Tomcat, a product of the Apache Software Foundation, was selected as the servlet engine because it is a free and open-source product. ArcIMS has been tested and certified to be compatible with several versions of Apache and Tomcat. The reader can find details on version compatibility at: http://support.esri.com/. *nowCOAST* Ver. 2 uses Tomcat version 5.5.4.

The Apache HTTP server and Tomcat must be configured to communicate with each other and to allow Apache to redirect requests for Java code execution to Tomcat. The redirect is accomplished by installing the mod_jk Apache module to the HTTP server and making modifications to the configuration of Apache. Information on the installation and configuration of mod_jk can be found on the Jakarta site (see below) as well as on the ESRI support site mentioned above.

Tomcat can be downloaded from http://jakarta.apache.org/. Additional background information is also available at that address.

For information about Java Servlets, see: http://java.sun.com/products/servlet/index.jsp
For information about JSP, see: http://java.sun.com/products/jsp/

2.3.3. Java Virtual Machine

Sun Microsystem, Inc.'s Java Software Development Kit (JDK) is the first requirement for the installation of ArcIMS and provides all the dependencies for running and developing Java applications. The Java Virtual Machine is a component of the development kit in which Java applications can be run. *nowCOAST* Ver. 2 uses J2SE 5.0 (ie Java 2 Standard Edition 5.0).

Sun's JDK can be obtained from: http://java.sun.com.

2.4. Schematic Diagram of *nowCOAST* Server Configuration

The *nowCOAST* server software described in Sections 2.2 and 2.3 is configured in a standard single-server setup, in which a single machine handles the Web server and servlet engine responsibilities, runs the ArcIMS software, and stores the data found in *nowCOAST*. In a multiple-server setup, the Web server and servlet engine may be running on one machine and ArcIMS on another.

The single-server setup was chosen because it is easier to configure and less expensive than a multiple-server setup. In addition, the projected load for the site was low enough that a single-server would be sufficient to handle *nowCOAST's* users.

Figure 1 shows a schematic view of how *nowCOAST's* server software is configured and the basic processing flow between the components when a user requests a map through the *nowCOAST* website. Most of the software components described in Sections 2.2 and 2.3 are shown in the diagram, except for the Java Virtual Machine (JVM). The JVM is responsible for running the Tomcat servlet engine, and essentially underlies Tomcat in the diagram.

The top part of the diagram shows the *nowCOAST* 'clients,' which consist of users' Web browsers or desktop software capable of communicating with ArcIMS, such as ESRI ArcGIS Desktop. The bottom half of the diagram shows the *nowCOAST* server and depicts how the software components relate and communicate with each other. More detail on some of the components and communication is included in following sections, including the *nowCOAST* Map Viewer application, how it works with Tomcat, and how Java Connector is used in the Map Viewer to communicate with the ArcIMS Application Server. In the diagram, the orange colored text (eg. 'HTTP/JSP') indicates some of the communication protocols that are being employed when the user requests a map from *nowCOAST*.

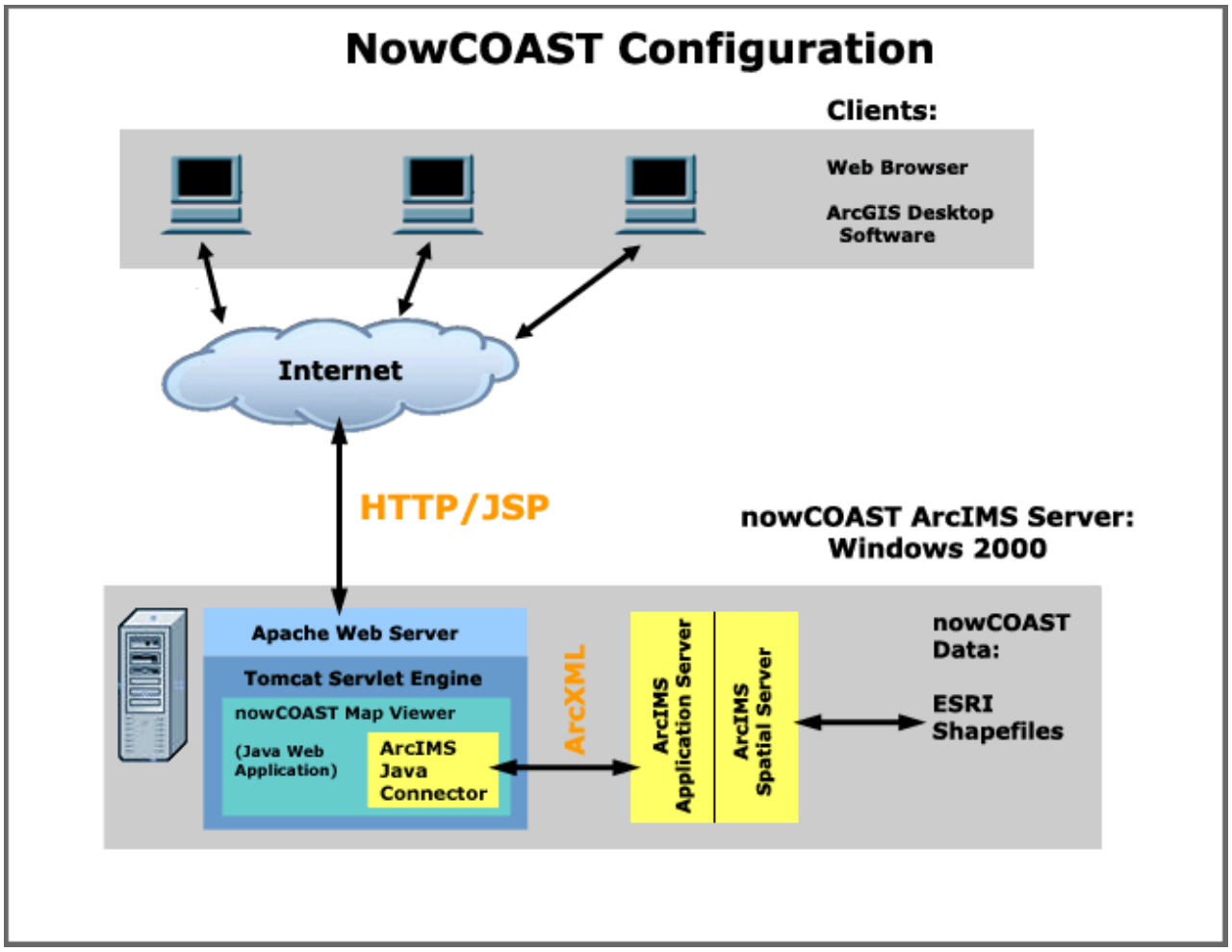

Figure 1. Diagram of the *nowCOAST* configuration server and the interaction that occurs with ArcIMS clients over the Internet.

3. INSTALLATION, CONFIGURATION, AND CUSTOMIZATION OF *nowCOAST* SERVER SOFTWARE

This section provides an overview of the steps taken in installing ArcIMS and the related software described in Section 2. For more detailed configuration information, the reader should review the Installation Appendix (Appendix A). Detailed documentation on installing ArcIMS is available on ESRI's website. The general procedure described in these installation documents was followed for *nowCOAST*. However several customizations were made which are discussed below and in the Appendix.

ArcIMS installation documentation can be found online at:
http://support.esri.com/index.cfm?fa=knowledgebase.techarticles.gateway&p=16&pf=1267

The above link contains a catalog of installation documents for the Apache Web server for Windows OS. The specific document used for *nowCOAST* depends upon the versions of Web server and Servlet engine software. See sections 2.3.1 and 2.3.2 for this information. ESRI recommends installing ArcIMS using only the specific Web server/Servlet engine versions described in each of these documents. However, ArcIMS can function using a number of version combinations.

nowCOAST uses a version of Tomcat that does not have installation documentation specific to it. Consequently, the reader should use the following Tomcat version 4.1 instructions available at: http://support.esri.com/index.cfm?fa=knowledgebase.techarticles.articleShow&d=24171

To replicate the exact *nowCOAST* configuration, follow instructions in the ESRI install document but install the JDK, Apache Web server, and Tomcat Web server versions described here. In addition, the user should follow the specific customization steps described below and in Appendix A.

3.1. Initial Steps for ArcIMS Install

First, a privileged local user is added to the *nowCOAST* server. This account is then used to perform the ArcIMS installation so that the ArcIMS processes will not be associated with a network or domain account.

3.2. Installing Non-ArcIMS Software

The following instructions mirror the steps in the ESRI install guide and include *nowCOAST*-specific requirements.

3.2.1. Java Development Kit

Download J2SE 5.0 from Sun Microsystems (http://java.sun.com). Install the JDK in a subdirectory named 'Java' under the operating system drive (e.g., C:/Java/jdk1.5.0).

3.2.2. Apache HTTP Server

Install the latest Apache 2 Web server, available at: http://httpd.apache.org. Install Apache under a 'Server' subdirectory (e.g., C:/Server/Apache2).

3.2.3. Jakarta Tomcat Server

Download the latest Tomcat 5.5 servlet engine from: http://jakarta.apache.org/tomcat/index.html. Install Tomcat under the same 'Server' subdirectory (e.g. C:/Server/Tomcat5.5).

3.2.4. Configuring Apache with Tomcat

1. Downloading of Apache Jakarta Tomcat Connector

Download the correct Apache Jakarta Tomcat Connector (mod_jk) from:
http://www.apache.org/dist/jakarta/tomcat-connectors/jk/binaries/win32/.

The above address is the online archive of current versions of mod_jk, where each mod_jk version corresponds to a specific version of the Apache 2 server. If the latest Apache 2 is being installed, then the latest mod_jk will be required.

2. Modifying the directory structure

The ESRI installation documentation provides two configuration files named 'mod_jk.conf' and 'workers.properties'. Use the two modified versions of these configuration files found in Appendix C instead of those provided by ESRI in this step of the Apache-Tomcat configuration process. The mod_jk.conf file and the workers.properties files are used to configure an Apache JServ Protocol (AJP) connector that allows Apache to talk to Tomcat.

If the directory structure for the Tomcat and the JDK installations differs from the directories specified in these instructions, the recommended directory names must be changed in the workers.properties file (see the ESRI install directions for details) to reflect the actual directory names used.

3. Modifying mod_jk.conf

The mod_jk.conf file is an important part of *nowCOAST's* server configuration. In the current *nowCOAST* server setup, Apache and Tomcat share responsibility for serving the *nowCOAST* map viewer content. Apache is configured to serve static content (images, HTML, JavaScript), whereas Tomcat serves dynamic content such as JSP pages. The mod_jk.conf file included above contains the Apache directives necessary for this configuration. However, certain entries must be changed to reflect the directory structure where the *nowCOAST* viewer is installed on the server. The Apache configuration file, httpd.conf, must also be changed to allow this configuration. Detailed

instructions on this step can be found in Appendix A.

4. Modifying Apache's config file – httpd.conf

Apache's httpd.conf file must be modified further than the ESRI install instructions specify. *nowCOAST* relies on server-side includes (SSIs) for some of its code. SSIs must be configured in the httpd.conf file in order to function. In addition, security changes must be made to Apache beyond what is specified in the ESRI install guide. Refer to Appendix A for step-by-step instructions for making the needed changes.

5. Modify Tomcat server.xml and web.xml files

Modifications must be made to two of Tomcat's configuration files, server.xml and web.xml in order for the *nowCOAST* viewer webapp to function correctly in this configuration. In Tomcat 5.5, additional changes, beyond those described in the ESRI install guide must be made to configure Tomcat's 'invoker' servlet in web.xml. Also, server.xml must be modified to set the *nowCOAST* webapp context to be the default Tomcat context. See Appendix A for detailed instructions on these changes.

6. Verify Apache/Tomcat configuration

Perform the "Hello World" example test as described in the ESRI install guide to verify that Tomcat and Apache have been configured correctly. If the test is successful, ArcIMS may be installed.

3.3. Installing ArcIMS

Install ArcIMS from the ESRI installation CD. Be certain that you are logged in on a local account with administrator privileges (not a network account) for this step, as this is critical to configuring ArcIMS properly.

When given the option for standard install or custom install, choose custom. The install dialog will then allow you to select several different packages to install with ArcIMS. Any of the examples and sample data may be installed. However, for *nowCOAST* to function properly, the Java Connector must be selected for installation using the Connectors menu. The Java Connector will be installed in the ArcIMS install directory from which it can later be copied to the *nowCOAST* webapp directory if needed.

Proceed with the ArcIMS installation once these customizations have been made.

3.4. ArcIMS Post Install and Verification

Follow the ESRI install guide for instructions on completing the ArcIMS post installation. There are a few different options available to configure in the post installation. Refer to Appendix A for more detail on selecting among these options for the *nowCOAST* install.

Once the post install is completed, run the diagnostic tests as described in the ESRI install guide. If successful, ArcIMS is installed and configured correctly.

3.5. Apache, Tomcat Security, Performance Tuning, and Logging

After ArcIMS has been installed correctly and tested, further tuning is necessary for security and performance of Apache and Tomcat for *nowCOAST*. Because *nowCOAST* has been designated as the default context in Tomcat, components that were installed by ArcIMS in Tomcat's ROOT webapp must be copied to the *nowCOAST* webapp directory. Additionally, *nowCOAST* requires performance tuning to the Java runtime environment in order to allow Tomcat to function correctly. Finally, further security enhancements should be made to Apache which involves modification of the httpd.conf file.

Full description of the changes in this step can be found in Appendix A, section 2.4.5.

The steps are numbered in the following order:

1. Copy ArcIMS components to the *nowCOAST* viewer webapp directory.
2. Performance tuning of the Java runtime environment for Tomcat.
3. Final Apache security measures.
4. Apache logfile rotation setup.

4. DEVELOPMENT AND MAINTENANCE OF THE *nowCOAST* MAPSERVICE

4.1. Purpose and Type

The ArcIMS MapService is the basic component of ArcIMS and is responsible for serving geographic data and images via the web to the user's web browser. A MapService in ArcIMS is configured by setting parameters in the ArcIMS Application Server. The main parameter that defines a MapService is the ArcIMS Map Configuration File, which is an ArcXML-based document of extension type '.axl' (hereafter referred to as AXL). The AXL file specifies everything from the location of geospatial data to be published on the server to user access control to the MapService if implemented. The AXL file is also responsible for specifying the content, order, and appearance/symbology of map data layers within the MapService, and any projection or other visual map specification desired. The configuration file can only be modified by the ArcIMS server administrator.

There are several types of ArcIMS services that can be published over the web from an ArcIMS server. The most commonly used types are MapServices which produce map images for web clients; however, there are several other types with different functionality such as: Feature Service, ArcMap MapService, or MetaData Service. *nowCOAST* implements a MapService to deliver content to its users because the project required an easy-to-use and relatively lightweight web browser-based client interface for users that accommodates simple view and query functionality. The MapService creates an image based upon user interaction with the interface and displays that image in the user's web browser. *nowCOAST* uses the Portable Network Graphics (PNG) image format since it is virtually patent free.

4.2. Initial Creation

The AXL file for the *nowCOAST* MapService was created initially using ArcIMS' Java-based editing tool, the Author application. Author has an interface similar to other ESRI GIS products and can be used to specify the basic aspects of an AXL file, allowing the user to define symbology (fill color, fill type, outline, etc.) of a map layer, map background, and some basic stored queries. For more elaborate customization of the AXL file, it must be edited using a text editor, as Author is somewhat limited in the scope of the modifications that it can make.

4.3. Modifications

Several modifications were made to the *nowCOAST* AXL file to achieve specific functionality required for *nowCOAST*.

4.3.1. Unlocked Shapefile Workspace

The AXL file was changed to allow modification of the shapefiles in the workspace while the MapService is running. The changes are depicted in bold in the statement below. If this change is not made, the MapService locks the shapefiles and must be stopped before any modifications are

allowed.

```
<SHAPEWORKSPACE name="shp_ws-0" directory="c:\ArcIMS\Shapefiles\nowcoast" shared="true" />
```

A drawback associated with enabling this feature is the risk of potential corruption of "live" shapefiles. This problem can be avoided by creating a test MapService and testing the shapefile before adding to the live MapService.

4.3.2. Scale and Value Dependent Rendering

Scale dependent rendering can be used to improve the appearance of dense feature datasets. The sample code from the AXL file seen below, specifies a particular feature display symbol size for the map layer *Weather/Ocean Stations*, which is dependent on the map scale. This code is a portion of a complete layer specification, which would include < *SCALEDEPENDENTRENDERER*> tags for other scale ranges. This code also uses the <GROUPRENDERER> tag to nest <SCALEDEPENDENTRENDERER> tags that vary symbol size depending on map scale. Within the <SCALEDEPENDENTRENDERER> tag is a <VALUEMAPRENDERER> tag that specifies only features that are marked as *'Avail = 1'* are rendered in the map.

```
<LAYER type="featureclass" name="Weather/Ocean Stations" visible="false" id="obs_pt_wo"
maxscale="1:20000000">
            <DATASET name="obs_pt_wo_20031215" type="point" workspace="shp_ws-0" />
              <GROUPRENDERER>
                <SCALEDEPENDENTRENDERER lower="1:16000000" upper="1:20000000">
                  <GROUPRENDERER>
                    <VALUEMAPRENDERER lookupfield="AVAIL" >
                      <EXACT value="1" label="Weather/Ocean Stations">
                        <SIMPLEMARKERSYMBOL color="153,0,0" type="star" width="6" />
                      </EXACT>
                    </VALUEMAPRENDERER>
                  </GROUPRENDERER>
                </SCALEDEPENDENTRENDERER>
                <SCALEDEPENDENTRENDERER lower="1:8000000" upper="1:16000000">
                  <GROUPRENDERER>
                    <VALUEMAPRENDERER lookupfield="AVAIL" >
                      <EXACT value="1" label="Weather/Ocean Stations">
                        <SIMPLEMARKERSYMBOL color="153,0,0" type="star" width="8" />
                      </EXACT>
                    </VALUEMAPRENDERER>
                  </GROUPRENDERER>
                </SCALEDEPENDENTRENDERER>
                  ……..
          </GROUPRENDERER>
</LAYER>
```

4.4. Updating

The two ArcIMS MapService components that can be updated are the underlying data (i.e. shapefiles) and the appearance (i.e. MapService AXL file). Updating the actual data consists of

copying shapefiles referenced in the live .axl file into the proper workspace or directory. An updated shapefile needs to be quality controlled prior to copying it to the workspace. This step avoids corrupting a live MapService. Updating the AXL file to change the appearance of an actual layer or to add new layers is completed by editing the AXL file with a text editor and then refreshing the MapService. The MapService can be refreshed either through ESRI's Java-based ArcIMS Administrator application or automatically updated with a script using ArcXML administration syntax. Changes made to an AXL file for a live MapService should normally be tested using a test MapService to avoid breaks in service of the live MapService resulting from syntax errors or other unwanted changes.

5. *nowCOAST'S* COMPONENTS: THE MAP VIEWER AND DATABROWSER

The *nowCOAST* user interface consists of two applications working in unison. The first is the map viewer, described earlier, which provides visualization of *nowCOAST's* underlying data. The second is the portal's 'Databrowser,' an interface that allows users to view and browse forward and backward through websites displaying the observations and forecasts that they locate through the map viewer. The following two sections describe the purpose, origin, and design of each application.

5.1. The Map Viewer

The interface through which the user experiences ArcIMS is the viewer. This viewer is a Web browser-based application that is downloaded to the client's browser and contains a collection of HTML, JavaScript and other code designed to communicate with ArcIMS. ESRI provides several standard templates for viewer applications, one of which is an HTML viewer.

nowCOAST replaces ESRI's standard HTML viewer with a custom-built JSP-based viewer that is adapted from an ESRI JSP template viewer. The JSP viewer is an improvement over the HTML viewer in that it provides better stability on the client machine. It also contains less JavaScript code making it a quicker download for low-bandwidth users. The JSP viewer was implemented for *nowCOAST* in Version 2.0.

5.1.1. Origin of Java Server Pages Map Viewer

The ESRI JSP viewer template on which *nowCOAST's* viewer is based is provided by ESRI as sample code for the Java Connector. It is intended to demonstrate some of the functionality that is incorporated into the Java Connector, but omits many aspects that can be added upon customization by the developer. Customizations for the *nowCOAST* viewer are discussed in the next section.

The JSP viewer communicates with the ArcIMS server software through the Java Connector, which is a Java-based server-side software package that translates the map request by the JSP viewer into ArcXML.

The JSP viewer differs from the HTML-based viewer in several ways. The main difference is that the JSP viewer communicates with the ArcIMS server software through the Java Connector, which is an add-on to ArcIMS. The JSP viewer uses the Java Connector to interpret the user's interaction with the viewer and to generate, on the server the ArcXML that is sent to the ArcIMS Application Server. In the HTML viewer, ArcXML creation must take place on the user's machine, which increases computational demand for the user. Moving this processing to the Java Connector on the server results in a reduction of code to download to the user's machine, resulting in quicker downloads and greater stability of the viewer for users.

From the user's perspective, there is little visual difference between the HTML and the JSP viewer. In theory, the two could be designed to look identical, and the difference in function would be entirely transparent to users, who might only notice performance differences. Like the HTML viewer, the JSP viewer contains HTML and JavaScript code, but it contains a fraction of the amount of the standard HTML viewer. The following sections discuss the design of the JSP viewer which uses JSP, JavaBeans, JavaScript, HTML and Cascading Style Sheets (CSS) code.

5.1.2. Development Goals

The development of *nowCOAST's* JSP viewer focused on duplicating most of the functionality that had been previously provided in the HTML viewer. These added functionalities include:

1) standard GIS tools to allow users to interact with the map and query *nowCOAST's* data,
2) four pulldown menus for novice GIS users,
3) an intuitive GIS layer list to allow users to easily find data they want,
4) a simple means of linking directly from a station on the map to a Web page displaying real-time information.

Many improvements on these basic aspects of *nowCOAST's* functionality were sought for the JSP viewer to accommodate the addition of future datasets.

5.1.3. Development of the 'Jumpbar'

nowCOAST was designed to serve both experienced and novice GIS and ArcIMS users. The Jumpbar interface was created to assist inexperienced users in locating data through *nowCOAST*. The Jumpbar is positioned above the map in the viewer. It consists of four Web-standard 'pulldown' menus (Fig. 2) which allow the user to specify:

1) location (seaport, coastal state, estuary, lake, marine sanctuary),
2) information (specified type of observation, forecast guidance, or forecast),
3) variable (water level, air temperature, wave height, etc.),
4) the time (specific forecast time or time period).

The Jumpbar was designed to visually guide the novice user through this four-step process to specify the criteria that they are interested in, then click on 'Go' to obtain a map with the requested information.

Figure 2. The *nowCOAST* 'Jumpbar'.

5.1.4. Development of the GIS Layerlist

The layerlist for the JSP viewer was redesigned from the standard type provided by ESRI template viewers to improve the functionality for *nowCOAST's* features. This layer list is seen in Figure 3. Two major functionality changes were made to 1) incorporate a legend graphic into the layerlist for each layer and eliminate the separate legend image generated by ArcIMS, and 2) add functionality to improve user control of gridded field display in the viewer. The layerlist code is based upon open source code written by Dave Bollinger (GIS Programmer/Analyst, San Joaquin County, CA) and distributed on the ESRI website at:
http://arcscripts.esri.com/details.asp?dbid=12191

1. Incorporation of Legend into Layerlist

The *nowCOAST* JSP map viewer Layerlist was designed to have merged legend information integrated with the layer visibility toggle functionality, which eliminated the need to show a separate ArcIMS generated legend graphic. It also provided the Layerlist functionality to the *nowCOAST* user as an alternate means to interact with the map from the Jumpbar. In the initial *nowCOAST* design, the Layerlist was not visible by default, and unless users were familiar with ArcIMS, they would not know that an alternate means to the Jumpbar was available. By merging legend information into the Layerlist, it is displayed by default in the JSP map viewer, so that users can easily see the additional layers that are available in the map.

2. Modification to Accommodate Gridded Fields

The JSP viewer for *nowCOAST* has been designed to incorporate the display of gridded fields from the National Weather Service's National Digital Forecast Database (NDFD) as well as from NCEP and NOS forecast models in the future. In the interest of simplifying this functionality for the user, the viewer was designed to allow for manipulation of gridded forecast fields from both the jumpbar and the Layerlist. An intuitive interface was sought for the Layerlist that would clearly show which gridded fields are available, which grid is currently displayed in the map view, and what the color-coded legend available represents.

The primary design criteria for the display of NOAA's gridded fields was to allow a user to easily browse the large number of fields. This was achieved by designing a dynamic Layerlist interface that allows a user to expand or collapse menus for each forecast variable of a NOAA gridded forecast product. With this dynamic interface, the user can choose to investigate only some fields of interest; other fields remain hidden within the Layerlist.

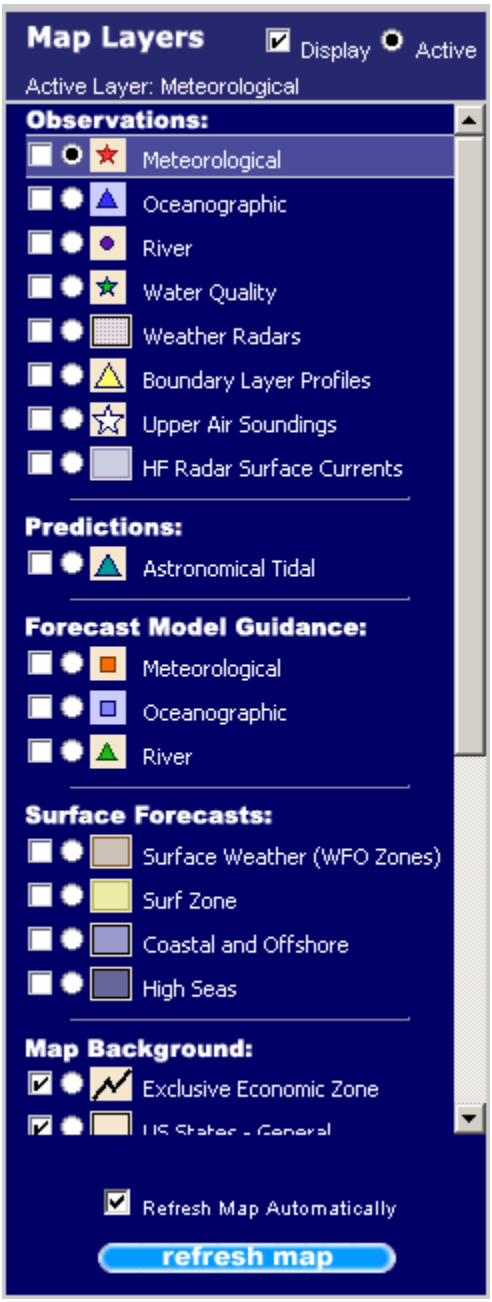

Figure 3. Screenshot of the *nowCOAST* map viewer Layerlist.

The Layerlist code is based upon Dynamic HTML (DHTML) principles and is written mostly in JavaScript. An effort was made to minimize the amount of JavaScript code involved since the JSP viewer itself is designed to use little JavaScript.

The Layerlist code was modified to prevent simultaneous display of gridded fields, as this would be confusing as well as create a potential drain on server resources. The DHTML for the gridded field section of the Layerlist allows only one layer in either the NDFD or forecast model output to be selected and therefore displayed. The corresponding legend information for that layer is also displayed in the viewer.

5.1.5. Development of GIS Tools

Several GIS tools that were required for *nowCOAST* had to be implemented from scratch as they were not provided as part of the JSP viewer template. These included:

- Zoom to Last Extent
- Query
- Print
- Identify
- Link to Data

Figure 4 shows the layout of the GIS tools that were implemented in the *nowCOAST* JSP map viewer. The custom tools are placed alongside other necessary map tools (e.g., Zoom In/Out, Pan, Select) that were adapted from the template JSP viewer on which the *nowCOAST* viewer was built. The tools are located in the map viewer window below the map image itself and are organized by function to allow easy selection by users and also to conserve space in the browser window. The section of the window above the tools contains an area with instructions on tool use as well as text displaying of the 'active' tool.

Figure 4. Screenshot of the *nowCOAST* map viewer map tools display.

5.2. The Databrowser

The Databrowser was created to allow users to better manage the Web pages with observations or forecasts that they link to through the map viewer. The Databrowser, like the map viewer, is based on JSP and therefore is easily packaged with the map viewer code and executed within Tomcat. When the user clicks on an observation station or forecast point or zone using the 'Link to Data' button in the map viewer, the Databrowser is launched in a second window to display the associated Web page. It also displays tabular data output from queries to *nowCOAST's* underlying data on observation stations and forecasts products using the 'Select by Rectangle', 'Identify', and 'Query' tools.

The Databrowser maintains a record of user interaction with *nowCOAST* throughout the duration of the user's session. When the user links to an observation station or forecast point or zone or

queries the underlying *nowCOAST* data, a record is added to the Databrowser with this information. Users are then able to browse back and forward through these records, delete an individual record, print any dataset returned from the *nowCOAST* database, or open a new window with only the live forecast or observation Web page. The Databrowser allows them to view this information within a single Web browser window instead of spawning a new window every time a new query is made to the viewer. This provides a needed user-interface improvement over *nowCOAST's* original design because it greatly reduces the number of open browser windows *nowCOAST* requires.

The Databrowser consists of a small Web browser frame with a set of buttons providing functionality to browse through the record history. Buttons included are: Previous Record, Next Record, Delete Record, Back to Map, and Open Link. Figure 5 illustrates the Databrowser layout.

Figure 5. Screenshot of the Databrowser layout.

6. DESCRIPTION OF *nowCOAST* LINKS TO COASTAL OBSERVATIONS

nowCOAST includes many distinct GIS map layers containing Web hyperlinks to information scattered across the internet. The first category of map layers included in *nowCOAST* relate to coastal observations. Described below are the types of coastal observation layers included in the site as well as general information about the criteria used in including coastal observation links in *nowCOAST*, the methods that were used to collect and maintain information about the links, and the process followed to convert the information into GIS formats.

6.1. Criteria

The criteria for including a link or links to an individual station or observing network are the following: 1) the data from the station are displayed reliably on a publicly-accessible web page in real- or near-real-time, 2) the web portal did not contain any commercial advertisement and offensive material, and 3) the data may be useful to the marine community.

6.2. Method of Collection

Information about the observing networks included in *nowCOAST* was compiled from existing data from the prototype portal and updated as needed. Additionally, improvements were made to the extent, accuracy, and organization of the data, along with the inclusion of additional observing networks that met the project criteria. Microsoft (MS) Access was used as the *nowCOAST* database for its ease of use in assimilating existing legacy data from a variety of sources such as MS Excel, DBase IV, and text. Much of the data were obtained directly from the network sponsoring agency or institution, either directly from the web or through correspondence. Technologies such as Perl and Structured Query Language (SQL) were utilized to manipulate the data.

6.3. Weather and Ocean Observation Sites

nowCOAST provides geo-referenced links to observations from federal-operated networks and non-federal networks. Information on these networks is given in Table 1.

6.4. River Observation Sites

nowCOAST provides links to the following real-time river observation networks:

- United States Geological Survey (USGS) NWIS Real-Time River Observation Network
- NOAA Integrated Flood Observation and Warning System (IFLOWS) Network

The USGS National Water Information System (NWIS) observation network consists of real-time observations of river gage height and associated streamflow as well as many meteorological and water condition and quality observations. There is nationwide coverage of these observations for most major rivers. The IFLOWS network has coverage for many states east of the Mississippi, particularly in the Mid-Atlantic region. IFLOW stations measure precipitation and in some cases record river stage information. See Table 1.

Information about the USGS NWIS network is available at: http://water.usgs.gov/waterwatch/
Information about the IFLOWS network is available at: http://www.afws.net/.

6.5. Water Quality Observation Sites

nowCOAST contains links to water quality observations from federal and state networks. Selected stations in the USGS NWIS real-time river flow network take water quality measurements. n*owCOAST* also links to observations from state networks including the following:

- UCONN EMPACT Network
- Maryland Dept. of Natural Resources (MDDNR) Network
- New Jersey Dept. of Environmental Protection (NJDEP) Network.

6.6. Doppler Weather Radars

nowCOAST includes links to 158 NWS NEXRAD Doppler weather radar sites located throughout the continental U.S., Puerto Rico, Alaska, and Hawaii.

6.7. Boundary Layer Wind Profilers

NOAA wind profilers and cooperative wind profilers, which measure wind speed and direction in the lower atmosphere, are included in *nowCOAST's* database of real-time observation links. *nowCOAST* provides links to both NOAA and cooperating wind profiler sites throughout the entire U.S.

These data are available on a centralized NOAA website at:
http://www.profiler.noaa.gov/npn/profiler.jsp.

Table 1. Information on observing networks linked to in *nowCOAST*.

Network Name	Abbreviation	Geographic Location	Type of Measurements	Agency or Ogranization	# of Stations
Federal government operated networks:					
Automated Surface Observing System	ASOS	Nationwide	Met[1]	NWS, DOD, FAA	825
Federal Aviation Administration's (FAA's) Automated Weather Observing System	AWOS	Nationwide	Met	FAA	151
Fixed buoys and Coastal-Marine Automated Network stations	C-MAN	Nationwide	Met Ocean	NWS	59
U.S. Climate Reference Network	CRN	Nationwide	Met	NOAA	69
Cooperative Integrated Flood Observing and Warning System	IFLOWS	Eastern U.S.	Met Riv[2]	NWS	2029
NOAA's Environmental Real-Time Observation Network	NERON	NY and New England	Met	NWS	102
Doppler Weather Radars	NEXRAD	Nationwide	Reflectivity	NWS and DOD	158
National Water Information System River Stream Gages	NWIS	Nationwide	Met and Riv	USGS	9196
National Water Level Observation Network	NWLON	Nationwide	Met and Ocean[3]	NOS	232
Physical Oceanographic Real-Time System	PORTS	Nationwide	Met Ocean	NOS	64
Upper-Air Atmospheric Soundings (radiosondes)	RAOBS	Nationwide	Met	NWS and DOD	36
National Interagency Fire Center's (NIFC) Remote Automated Weather Stations	RAWS	Nationwide	Met	NWS	1495
GLERL's Real-time Meteorological Observation Network	RTMON	Great Lakes Region	Met	NOAA	8
DCNet/URBANet	UrBANet	DC/MD/NY	Met	NOAA/OAR	10
Non-Federal government operated networks:					
UCAR 4DWX Weather Observing network	4DWX	Chesapeake Bay	Met	DOD	4
University of New Hampshire's AIRMAP network	AIRMAP	NH	Met	OAR	3
State and University Operated Boundary Layer Wind Profilers	BLP	Nationwide/Canada	Met	NOAA	65
Bodega Marine Laboratory	BML	CA	Met Ocean	University of California Davis	1
Carolinas Coastal Ocean Observing and Prediction System	Caro-COOPS	SC	Met Ocean WQ[4]	USC, NCSU, UNCW	8
Univ. of MD's Chesapeake Bay Observing System	CBOS	Chesapeake Bay	Met Ocean	UMD	3
Columbia River Estuary	CORIE	Oregon	Met Ocean	Oregon Health & Science University	17

[1]Meteorological [2] Oceanographic [3] River [4] Water Quality

Network Name	Abbreviation	Geographic Location	Type of Measurements	Agency or Ogranization	# of Stations
Environmental Monitoring for Public Access Community Tracking	EMPACT	Long Island Sound	Met Ocean WQ	UCONN	8
Gulf of Maine Observing System	GoMOOS	Maine	Met Ocean	NDBC	11
GPS Surface Observing System	GSOS	Nationwide	Met	FSL/DOD	72
High-Frequency Surface Current Radars	HFR	Nationwide Coastal Regions	Ocean	Universities	11
Innovative Coastal-Ocean Observing Network	ICON	CA	Met Ocean	Monterey Bay Aquarium Research Institute	3
Louisiana Agriclimatic Information System	LAIS	LA	Met	LSU	25
Rutger's LEO-15	LEO	NJ	Ocean	Rutger's University	2
Long Island Shore Network	LIShore	NY	Met Ocean	USACE	5
Louisiana Universities Marine Consortium	LUMCON	LA	Met Ocean WQ	Louisiana Universities Marine Consortium	5
Maryland's Department of Natural Resources water quality monitoring station network	MDDNR	MD	Ocean WQ	MDDNR	18
Maryland's DOT automated meteorological station network	MDOT	MD	Met	MDDOT	55
Moss Landing Marine Lab	MLML	CA	Met	Cal-STATE	1
Martha's Vineyard Coastal Observatory	MVCO	MA	Met Ocean	NWS	3
North Carolina Coastal Ocean Observing System	NCCOOS	NC	Met	NDBC	1
Network for Environmental Observations of Coastal Ocean	NEOCO	CA	Ocean WQ	University of California	7
New Jersey Coastal Monitoring Network	NJ CMN	NJ	Met Ocean	Steven's Institute of Technology	3
New Jersey DEP (Deptartment of the Environment) Water Quality	NJDEP	NJ	Ocean WQ	NJ DEP	4
Prince William Sound Observing Network	PWSN	AK	Met	OSRI	7
South Atlantic Bright Synoptic Offshore Observational Network	SABSOON	GA	Met Ocean	Skidaway Institute	3
Scripps Institution of Oceanography	SCRIPPS	CA and HI	Ocean	NDBC	18
Coral Reef Early Warning System	SEAKEYS-CREWS	Caribbean	Met Ocean	NOS	2
Stevens' Institute of Technology	SIT	NJ	Met Ocean	NDBC	3
Texas Automated Buoy System	TABS	TX	Ocean	Texas A&M	9
Texas Coastal Observation Network	TCOON	TX	Met Ocean	Texas A&M	48
University of South Florida's Coastal Observing and Prediction System	USF COMPS	FL	Met Ocean	NDBC	12
Vermont Monitoring Cooperative's Lake Champlain Observing System Network	VMC	VT	Met and Riv	VMC &NWS WFO Burlington, VT	2
Washington Department of Transportation's Washington State Ferries observing network	WADOT	WA	Met	WADOT	6
Wave-Current-Surge Information System	WAVCIS	LA	Met Ocean WQ	Louisiana State University	7
TOTAL:					**14886**

6.8. Upper-Air Sounding Stations

nowCOAST contains links to graphical displays of NOAA's Upper-Air Sounding stations. The upper-air data are collected on a twice daily basis at many stations throughout the U.S. *nowCOAST* provides links to graphical plots from active sounding stations throughout the coastal United States. The upper-air sounding data are available at several federal and educational websites; however, *nowCOAST* currently links to the UCAR Research Applications Program site at http://www.rap.ucar.edu/weather/upper/.

6.9. High Frequency Surface Current Sensors

nowCOAST links to surface current observations from high frequency (HF) radar networks in select coastal areas. Currently there are about 10 HF radar networks that *nowCOAST* links to.

6.10. Conversion to Shapefiles

All observation datasets are converted to ESRI shapefile format before they are brought into ArcIMS. Because each of the observation layers is a point shapefile (i.e., each observation station is treated as a point), this process consists of exporting the data from the database and converting to shapefile format using ESRI ArcGIS or ArcView software. While this process could be automated in the future, it is now done manually on an as-necessary basis. Details about the current procedure used for shapefile conversion can be found in Appendix B.

7. DESCRIPTION OF *nowCOAST* LINKS TO NOAA'S FORECASTS AND PREDICTIONS

In addition to links to coastal observations, the second category of links included in *nowCOAST* relate to NOAA text-based point forecasts and predictions. This section outlines the criteria used for inclusion of links to forecasts and predictions in the site, descriptions of individual types of forecasts and predictions and the sub-types included with each, as well as general information relating to the conversion of the collected data to GIS formats.

7.1. Criteria

In addition to providing links to real-time environmental information, *nowCOAST* was created to provide access to NOAA forecasts and forecast model guidance. This includes general text forecast products such as NWS County Forecasts and Marine Forecasts as well as more technically geared products such as NWS Model Output Statistics (MOS) and point guidance from NOAA forecast models such as WaveWatch III and Extra-Tropical Storm Surge (ETSS). The criteria for inclusion were any publicly web-accessible NOAA forecast product that is of use to the maritime and coastal community and/or the coastal research community.

7.2. Method of Collection

Collection methods for NOAA forecasts and forecast model guidance were similar to the techniques used for real-time weather and other environmental condition data (See Section 6.2.).

7.3. Weather and Ocean Model Forecast Guidance

nowCOAST provides links to model forecast guidance from several of NOAA's weather and ocean prediction models. These sources are listed in Table 2.

Table 2. Weather and ocean forecast model guidance linked to in *nowCOAST*.

Forecast Model Name	Abbreviation	Agency or Ogranization	Geographic Location	Forecast Variable	# of Stations
Model Output Statistics	MOS	NWS/MDL	Nationwide/ Southern Canada	Met	1731
Extra-Tropical Storm Surge Model	ETSS	NWS/MDL	East Coast, GulMex, AL	WL (water levels)	77
WaveWatch III Model[5]	Wave Watch III	NWS/NCEP	Nationwide	Wave Info	6
Chesapeake Bay Operational Forecast System	CBOFS	NOS	DE/MD/VA	WL	11
Port of New York/New Jersey Forecast System	NYOFS	NOS	NY/NJ	WL, Currents	8
Galveston Bay Operational Forecast System	GBOFS	NOS	TX	WL,WT (Water temperature), Salinity, Currents	6
Advanced Hydrologic Prediction Service	AHPS	NWS/RFCs	Nationwide	Stage, Discharge	3388
TOTAL:					**5227**

[5]Abbreviations for the WaveWatch III oceanographic model guidance geographic region types:

NWW3 global 1x1.25 degr (Global)	G
Alaskan Waters Model	AW
Western North Atlantic Model	WNA
North Atlantic Hurricane Model	NAH

7.4. River Model Forecast Guidance

nowCOAST provides links to river stage forecast guidance generated by forecast models at NOAA's 13 regional River Forecast Centers (RFCs). Web display of river stage forecast guidance varies at present from one geographic region to another, and some stations do not yet have forecasts implemented. In such cases, a time series graphic showing stage observation will often be substituted for a forecast. Stations within certain geographic regions are considered part of the NWS'Advanced Hydrological Prediction Service (AHPS). More information about AHPS can be found below:

Information about RFC river forecasts and AHPS is available at:
http://www.crh.noaa.gov/ahps/
http://www.srh.noaa.gov/default1.html.

7.5. NWS Zone Weather Forecasts

nowCOAST provides links to NWS Weather forecasts for zones throughout the U.S. The zones usually correspond to counties or sub-county areas. These links are created for the user selected NWS zone (i.e. WVZ053). Depending on the zone selected, the forecast is either a static page that is updated four times per day or a NDFD point forecast which is updated as frequently as every hour. The resulting forecast web page contains a 7-day prediction of meteorological parameters such as high and low air temperature, wind speed and direction, 12-hour probability of precipitation, 6-hour precipitation amount, and the type of sensible weather (i.e., rain, snow, fog, etc.). The zone forecast page also contains current weather conditions and a link to the appropriate satellite and radar image. Zone forecast data are currently available for the continental U.S., Puerto Rico, and Hawaii.

More information about NWS zone forecasts is available at the following web address:
http://weather.gov/ndfd/ and http://www.nws.noaa.gov/.

7.6. Surf Zone Forecasts

nowCOAST links to NWS WFO's surf zone forecasts for selected beach areas throughout the coastal U.S. wherever the forecasts are produced. Coverage exists currently on the Atlantic coast from New Jersey southward, parts of Gulf Coast Florida and Texas, and in the Pacific for Southern California and Hawaii. Some forecasts are seasonal only depending on location.

7.7. Marine Weather Forecasts

7.7.1. Coastal and Offshore

nowCOAST links to NWS WFO's coastal marine forecasts and NWS/NCEP Ocean Prediction Center's offshore marine forecasts. These forecasts are critical information sources for mariners and the coastal community.

7.7.2. High Seas

nowCOAST links to NWS/NCEP Ocean Prediction Center's high seas forecasts. As in the Coastal and Offshore Forecasts, these forecasts are critical information sources for the maritime community. The High Seas Forecast shapefile was downloaded from the NWS website and then modified by Perl script to include URL information for *nowCOAST* hyperlink.

7.8. Astronomical Tidal Predictions

nowCOAST provides links to NOS astronomical tidal predictions available at 277 coastal U.S. sites from NOS' Center for Operational Oceanographic Products and Services (CO-OPS). These predictions are based on astronomical tidal forcing and are generated for tidal observation stations throughout the coastal U.S.

More information about these stations is available at:
http://tidesandcurrents.noaa.gov.

7.9. Conversion to Shapefiles

Each forecast dataset is converted into ESRI shapefile format before it is brought into ArcIMS. Unlike the observation datasets, some forecast datasets are polygonal in nature (NWS regional weather forecasts and marine forecasts). The process for conversion is similar to that of point datasets. However, the shapefile .dbf must be imported to a database and modified to include necessary fields such as URL and then re-exported back to the shapefile. This step allows for retention of the original shape information in the shapefile. More information about this process can be found in Appendix A.

Specific modifications were performed on some forecast datasets included in *nowCOAST*. The surf zone forecasts were generated by merging features from the NWS Zone Weather Forecast shapefile for areas where surf zone forecasts links are provided in *nowCOAST*. The resulting shapefile therefore depicts these merged areas. The Coastal and Offshore Forecast layer in *nowCOAST* consists of two NWS shapefile datasets that were merged together to unify the Coastal and Offshore regions. Both shapefiles were downloaded from the NWS repository, some zones in the Coastal forecasts shapefile were removed for areas that overlapped with Offshore zones (and for which no Coastal forecasts are provided), and the merged shapefile was populated with the necessary URL information described above. The coastal zones, which were removed, are located on the East Coast, north of Florida to Maine.

8. DESCRIPTIONS OF *nowCOAST'S* STATIC MAP LAYERS

The following sub-sections describe the static or map background GIS layers that are included in *nowCOAST*. To supplement the layers of hyperlinks included in *nowCOAST* these additional layers were added to give proper context and geographic reference. The layers are grouped below by similar content-types.

8.1. Hydrology

8.1.1. U.S. Coastal Watersheds

nowCOAST contains a hydrologic unit boundary shapefile for coastal U.S. watersheds. The shapefile in *nowCOAST* is a subset of a national watershed shapefile obtained from the U.S. National Atlas. *nowCOAST* displays hydrologic unit areas which are adjacent to the coastline.

Information about the National Atlas is available at: http://www.nationalatlas.gov/
Information about the hydrologic unit source shapefile is available at:
http://www.nationalatlas.gov/mld/hucs00m.html.

8.1.2. U.S. Rivers

nowCOAST includes a major U.S. river shapefile to show most coastal rivers and major navigable and non-navigable inland rivers. The rivers shapefile was obtained from the National Atlas.

Information about the rivers shapefile is available at
http://www.nationalatlas.gov/mld/hydrogm.html.

8.1.3. U.S. Lakes

nowCOAST includes a major U.S. Lakes shapefile as a background to provide geographic context and information for many of the non-coastal real-time weather and environmental observation links. The lakes shapefile was obtained from the ESRI Data & Maps CD distributed with ArcGIS software and the original source for the data is ArcWorld data.

Information about datasets available from ESRI can be found at: http://www.esri.com/data/.

8.1.4. World Lakes

A World Lakes shapefile is included in *nowCOAST* as a background layer. The data were obtained from the ESRI Data & Maps CD and originated from ArcWorld data. More information about the ESRI Data & Maps CD and ArcWorld data is available at the same link as U.S. Lakes (section 8.1.3).

8.2. Population

8.2.1. U.S. Cities

The U.S. Cities point shapefile in *nowCOAST* encompasses all cities of population greater than 5000. The data originated from the National Atlas and were modified to include only cities with a population larger than 5000. Scale dependent rendering is employed in the *nowCOAST* MapService to display cities of lesser and lesser population as map scale is decreased to avoid map clutter at larger scales.

Information about the cities shapefile is available at:
http://www.nationalatlas.gov/mld/citiesx.html.

8.2.2. U.S Urban Areas

nowCOAST includes a polygonal Urban Areas shapefile to add further context to point-based real-time weather and environmental observations and NOAA forecasts. The data are from the National Atlas Urbanized Areas shapefile from the National Atlas 'Digital Chart of the World' of 2001. As with the cities shapefile, scale dependent rendering is used to display urban areas of lesser and lesser population as map scale is decreased.

Information about the urban areas shapefile is available at:
http://www.nationalatlas.gov/mld/urbanap.html.

8.3. Transportation

nowCOAST includes two shapefiles of road information for the U.S.: Interstates and Major Roads. Scale dependent rendering is utilized so that at larger scales the Interstates shapefile is displayed and at smaller scales the Major Roads shapefile is displayed. Both the major roads and interstates shapefiles were obtained from the ESRI Data & Maps CD.

8.3.1. U.S. Interstates

The Interstates shapefile is from U.S. National Transportation Atlas Interstate Highways shapefile, published by the Bureau of Transportation Statistics in 2000. Additional information may be found at the ESRI site.

8.3.2. U.S. Major Roads

The Major Roads shapefile is from the National Atlas Roads shapefile published by the USGS in 2002.

Information about the major roads shapefile is available at:

http://www.nationalatlas.gov/mld/roadtrl.html.

8.4. Political/Legal Boundaries

8.4.1. States

nowCOAST contains a U.S. States shapefile displaying political borders within the United States. The U.S. States shapefile originated from the National Atlas.

Information is available at: http://www.nationalatlas.gov/mld/statesp.html.

8.4.2. International

nowCOAST contains a shapefile of international political borders. Depicted are all world countries with optional name labeling. The source of the shapefile is the ESRI Data & Maps CD, with the data originating from the ArcWorld Supplement. See the ESRI data website for additional source information.

8.4.3. Exclusive Economic Zone

A depiction of the legal boundary of the U.S. Exclusive Economic Zone (EEZ) is provided in *nowCOAST* as a reference layer. The EEZ shapefile was obtained from NOS' Office of Coast Survey in several individual regional shapefiles and was combined to form a single EEZ shapefile for the Continental U.S. and Puerto Rico. The original points used to create the polyline shapefiles were taken from the Federal Register, August 23, 1995, "Exclusive Economic Zone and Maritime Boundaries: Notice of Limits."

More information about the EEZ shapefile is available at:
http://www.nauticalcharts.noaa.gov/csdl/eez.htm.

8.4.4. National Marine Sanctuaries

nowCOAST contains a shapefile depicting boundaries of U.S. National Marine Sanctuaries (NMS) for reference purposes. The shapefile was obtained from the NOS National Marine Sanctuaries office, a division of the Office of Ocean and Coastal Resource Management.

More information about NMS can be found at: http://www.sanctuaries.nos.noaa.gov/.

9. METADATA FOR *nowCOAST* LAYERS

Metadata including URL addresses on Web sites providing real-time observations and NOAA forecast products were collected manually with a series of Perl scripts and inserted into a MS Access database. Quality control, verification, and conversion scripts were also written in Perl and connected to the Access database via the Open Database Connectivity (ODBC) Perl Database Interface (DBI) module. Dbase IV (DBF) files were exported from Access and imported into ESRI's ArcView 3.x. The DBF tables were then converted to shapefiles and added as content to the ArcIMS MapService. Additional information on this process is available in Appendix B.

The Federal Geographic Data Committee (FGDC)-compliant metadata for *nowCOAST* are available at the National Spatial Data Infrastructure Clearinghouse at: http://www.fgdc.gov/clearinghouse/clearinghouse.html.

The metadata are also available at: http://www.nauticalcharts.noaa.gov/csdl/op/ncoast_met.html.

The metadata refer to each online source of information included in the *nowCOAST* Web portal.

10. *nowCOAST* AND SECTIONS 508 AND 515

10.1 Section 508

The regulations implementing The Rehabilitation Act of 1973 Amendments (Section 508) have been adopted as a U.S. Department of Commerce standard. According to NOAA's Office of the Chief Information Officer, the Section 208 regulations require that individuals with disabilities who are members of the public or Federal employees, who are seeking information or services from a Federal agency, have access to and use of information and data that are comparable to that provided to the public generally or to other or Federal employees (respectively), who are not individuals with disabilities, unless an undue burden would be imposed on the agency (Steinborn, 2005, personal communication).

According to the NOAA Office of the Chief Information Officer (CIO), second to the law itself, the authority on all matters related to Section 508 is the U.S. Access Board, an independent Federal agency. The Board is structured to function as a coordinating body among Federal agencies and to directly represent the public, particularly people with disabilities. The Board's Web site
(http://www.access board.gov/sec508/standards.htm) lists Section 1194, which implements Section 508 for activities of Federal agencies in developing, procuring, maintaining, or using electronic and information technology including Web sites. Section 1194.3(e) states that "this part shall not be construed to require a fundamental alteration in the nature of a product or its components." Elsewhere "fundamental alteration" has been interpreted to mean a change in the fundamental characteristic or purpose of the product or service, not merely a cosmetic or aesthetic change. Thus, adding access should not change the basic purpose or characteristics of a product in a fundamental way (Steinborn, 2005, personal communication).

nowCOAST is a Web portal that centralizes a variety of links to real-time data hosted by other NOAA and non-NOAA entities. Its main purpose is to act as a central site for real-time data access. All data found in *nowCOAST* are created by other parties. Presently, it is not possible to change *nowCOAST's* map-based interface to make the portal's content accessible to persons with visual disabilities without fundamental alterations. This makes the site an exception under 1194.3(e). An explanatory HTML "alt-tag" has been added to explain the purpose and nature of the site and its exception under Section 508 when reader software "views" *nowCOAST* for the first time during a session.

As assistive technology products and web mapping technology advances *nowCOAST* developers will attempt to improve *nowCOAST* to make it accessible to the broadest possible audience including people with disabilities according to Section 508. In the meantime, the *nowCOAST* developers will support any data requests or assistance in accordance with Section 508.

10.2 Section 515

NOAA, in response to The Information Quality Act (Section 515), has issued Information Quality

Guidelines for ensuring and maximizing the quality, objectivity, utility, and integrity of information which it disseminates (http://www.cio.noaa.gov/itmanagement/infoq.htm and http://www.noaanews.noaa.gov/stories/iq.htm). The *nowCOAST* web mapping portal includes geo-referenced links to real-time observations from non-NOAA networks originally posted on non-NOAA web pages. According to the NOAA guidelines, the information not covered by these guidelines includes "hyperlinks to information that others disseminate, as well as paper-based information from other sources referenced, but not approved or endorsed by NOAA." In response to Section 515, the *nowCOAST* developers have included a pop-up window to caution users (when they first request observations) that the reliability, quality, and representativeness of observations from non-NOAA networks can vary by station or observing network. A similar disclaimer is included on *nowCOAST's* "Info Depot" Web page as well.

The *nowCOAST* portal links to operational or experimental forecast products from NOAA's, national and regional forecast facilities. These products are generated and disseminated primarily by NWS and NOS. NWS and NOS forecast facilities make every effort to ensure the quality, utility, objectivity, and integrity of their atmospheric, estuarine, marine, oceanographic, river, ecological, and/or air quality forecasts according to Section 515.

11. MONITORING AND UPDATING *nowCOAST'S* LINKS AND MAP LAYERS

11.1. Perl-Based Script to Check for Active Links

The *nowCOAST* web portal's database of real-time links to weather observations and forecasts must be continually monitored to ensure that links are correct and up-to-date. These links number in the thousands and can be altered at any time by the institutions that provide them. In order to maintain accuracy, a custom Perl script was designed to test each link via Hypertext Transfer Protocol (HTTP) GET requests and catalog server responses, alerting operators of any dead links or inconsistencies. This script is scheduled to run on a weekly basis.

nowCOAST's data are stored in a Microsoft Access database, from which GIS shapefiles are generated for display on the site. The Perl script iterates through the database to obtain the web address of each real-time weather observation station or forecast point, sends an HTTP GET request to the remote server, and then analyzes the server's response. Responses that indicate problems with the URL are cataloged and reported in a log file. The script is dependent upon a parameter file that specifies the Access table names to check.

The first step of this process, obtaining information such as URL, Station ID, and Station Name for each record in the database, is accomplished through SQL query of the Access Database and parsing of the query response.

These results are assigned to variables which are then passed to a subroutine that creates an HTTP GET request, opens a TCP connection to the remote server, sends the request, and then examines the response for 'success' or 'error' response codes.

Depending upon the result, the script will write either to an error output file (nowCOAST_errorlog.txt), or a comprehensive output file (xx_results.txt where 'xx' is an Access table name). The output files are written using standard Perl formats.

A generalized table of HTTP response codes and the resulting action that should be taken to correct the problem is found below. In general, a 200 response code indicates that the HTTP request was successful. Numbers above 300 indicate some error or change has occurred with the link, and the database should be updated accordingly. The most common error response code, 404 Not Found, indicates that the link is no longer valid and for *nowCOAST* purposes, a new link must replace it or the station or forecast should be deactivated in the Mapservice.

Table 3. Potential HTTP response codes and corresponding actions for *nowCOAST* operators.

HTTP RESPONSE CODE	ACTION
100	No Action Required
100 Continue	
200 - Success	200 response codes require no action and are not included in the error log file.
200 OK	
201 Created	
202 Accepted	
300 - Redirection	300 response codes indicate some sort of URL redirection and require update of database to include the new URL. This involves opening the URL in a web browser and recording the new URL
300 Multiple Choices	
301 Moved Permanently	
302 Moved Temporarily	
400 - Error	400 response codes indicate a bad URL. This is either due to an error in the URL structure or more often a discontinuation of data dissemination from the station in question. The operator should check to be sure that the URL does not contain a typo and if not set the AVAIL flag in the MS Access database to 0, indicating that the station is unavailable.
400 Bad Request	
401 Unauthorized	
402 Payment Required	
403 Forbidden	
404 Not Found	
405 Method Not Allowed	
406 Not Acceptable	
500 - Error	500 response codes indicate a server or HTTP error, URL in question should be tested manually
500 Server Error	
501 Not Implemented	
505 HTTP Version not supported	
Custom Response Messages	
STATION NOT FOUND	Station not displaying data, Set Avail = 0
NOT RESPONDING	Server Hanging on HTTP Request, Test URL Manually

The script is scheduled to run once a week on Sunday morning. Total execution time of the script is about six hours. The log file created by the script is located on ocs-w-mwengren. .

11.2. Including Links to New Observations or NOAA Forecasts

11.2.1. Modification of Access Database

Modification to *nowCOAST's* MS Access database is performed on an as-needed basis. Depending upon additions of new data or the results of the Perl-based link check script, changes

can be made to *nowCOAST's* data. All changes must be accompanied by shapefile creation and ArcIMS incorporation (see 10.2.2).

12. ACKNOWLEDGEMENTS

The project is supported by NOS' Coast Survey Development Lab and is funded in part by a grant from the NOAA High Performance Computing and Communications (HPCC) Office. The initial development of *nowCOAST* was funded by a grant from NOAA/National Environmental Satellite, Data, and Informational Service's Environmental Services Data and Information Management (ESDIM) Program. The portal seeks to support one of ESDIM's program objectives, which is to "improve access to NOAA environmental data and information for scientists and decision makers."

The portal would not have been possible without the cooperation of various individuals and groups within and outside of NOAA in providing information about their observing stations and networks and forecast sites or zones and posting real-time information on their web pages.

Thanks are expressed to the CSDL personnel who reviewed the report and provided comments.

APPENDIX A. *nowCOAST* SERVER INSTALLATION

This appendix contains detailed instructions for installing ArcIMS and the necessary Web server/servlet engine configuration for *nowCOAST*. Step-by-step instructions are given below for modifying the necessary configuration files. Numbers refer to the sections in the main document.

A.1. Configuring Apache with Tomcat

Step 1. mod_jk.conf:

Modify paths in mod_jk.conf: the mod_jk configuration file to reflect the directory where the *nowCOAST* viewer Web application (webapp) is installed. In the preliminary file included with this document, the viewer is installed in the 'C:/Shared/nowcoast_jspbeta/build/' directory. Although the viewer code can be placed anywhere on the file system, Tomcat and Apache need to know where it is located, so if it is installed elsewhere, the paths in mod_jk.conf must be changed accordingly.

- mod_jk.conf contains Apache VirtualHost directives for both of *nowCOAST's* domain names (nowcoast.noaa.gov and nowcoast.ncd.noaa.gov). Within each of these, there are several directives that contain the path to the *nowCOAST* webapp location on the file system. Each of these must be changed to the correct path (the root directory of the webapp).

  For example (under first VirtualHost directive):
  `DocumentRoot "C:/Shared/nowcoast_jspbeta/build"`

  Should be changed to:
  `DocumentRoot "C:/path/to/nowcoast"`

- Change all of the paths to the correct *nowCOAST* webapp location. Some of the Directory directives point to the /help subdirectory within the *nowCOAST* webapp. Maintain the /help subdirectory beneath whatever the new root directory is for these cases.

  `C:/path/to/nowcoast/help`

Step 2. Apache httpd.conf:

Apache must be told to load the mod_jk.conf file edited in the previous section. Instructions for this step are provided in the ESRI install guide. However instead of adding the Include directive in the guide, add the following lines to the bottom of httpd.conf (where `CATALINA _HOME` is the Tomcat install directory and the value that was set for the catalina_home environment variable).

```
NameVirtualHost *:80

Include <CATALINA_HOME>/conf/mod_jk.conf
```

The next step is to enable SSIs in Apache. Find in httpd.conf the AddType directive. Uncomment the line with the AddType directive as well as the AddOutputFilter line so they look as follows:

```
AddType text/html .shtml
AddOutputFilter INCLUDES .shtml
```

Additional security steps must also be added to the httpd.conf file. Find in the file the DocumentRoot parameter. The default DocumentRoot will point to the Apache 'Welcome' screen (as tested in the earlier part of the ArcIMS install). It is okay to leave this unchanged, as the VirtualHost directives in mod_jk.conf will override the setting. Below the DocumentRoot, there should be a Directory directive that looks similar to:

```
<Directory />
    Options FollowSymLinks
    AllowOverride None
    Order allow,deny
     Allow from all
</Directory>
```

Change these values to the following to increase the default security settings for Apache:

```
<Directory />
    Options FollowSymLinks
    AllowOverride None
    Order allow,deny
    Deny from all
</Directory>
```

The changes to the default Directory directive instruct Apache to set a restrictive set of features by default. Other Directory directives may override these settings, but these provide a basic security configuration for Apache.

Step 3. Modify Tomcat server.xml and web.xml files

Open the web.xml file in Tomcat's conf directory (<CATALINA_HOME>/conf/web.xml). Follow the instructions in the ESRI install guide to uncomment the 'invoker' servlet mapping. Also, because the invoker servlet definition is disabled by default in Tomcat 5.5, you must uncomment the servlet definition in web.xml. This section is near the top of the web.xml file. Search for the phrase "invoker" in the file and uncomment the following XML tags below the description of the invoker servlet:

```
<servlet>
    <servlet-name>invoker</servlet-name>
    <servlet-class>
        org.apache.catalina.servlets.InvokerServlet
        </servlet-class>
```

```
        <init-param>
           <param-name>debug</param-name>
           <param-value>0</param-value>
        </init-param>
    <load-on-startup>2</load-on-startup>
</servlet>
```

This will instruct Tomcat when it is reloaded to start the invoker servlet and accept incoming requests with the extension "/servlet" which is necessary for ArcIMS to function properly.

Open the server.xml file in Tomcat's conf directory (<CATALINA_HOME>/conf/server.xml). The server.xml file is the main Tomcat configuration file that specifies the behavior of the Tomcat servlet engine when it is started. Modifying the contents of the server.xml file allows a developer to change aspects of Tomcat such as the main directory in which webapps are placed as well as the permissions and behavior of webapps that are loaded in Tomcat. For the *nowCOAST* configuration, it is necessary to assign the *nowCOAST* webapp as the default context. To do this find within the server.xml file the first occurrence of the <host….. > element. The host element will contain various attributes and should contain `name="localhost"`, designating it as the default localhost host. Beneath this host tag, comment out the following line with the predefined default context so it looks as follows:

```
<!-- <Context path="" docBase="ROOT" debug="0"/> -->
```

Next, add the following context tag below the line that was commented out:

```
<Context path="" docBase="C:/path/to/nowcoast"
reloadable="false" useNaming="false" debug="0"
privileged="false" swallowOutput="false">
```

If necessary, replace the path with the directory where the *nowCOAST* webapp is installed on the server (as in previous steps). Changing these context elements allows the *nowCOAST* webapp to be the default context, accepting all requests to Tomcat which are not assigned to other contexts.

A.2. ArcIMS Post Install and Verification

Following are step-by-step instructions for the ArcIMS post install process.

When the post install dialog comes up, select the 'Custom' option for post install. On the subsequent screen be sure to select the options highlighted in Figure 6.

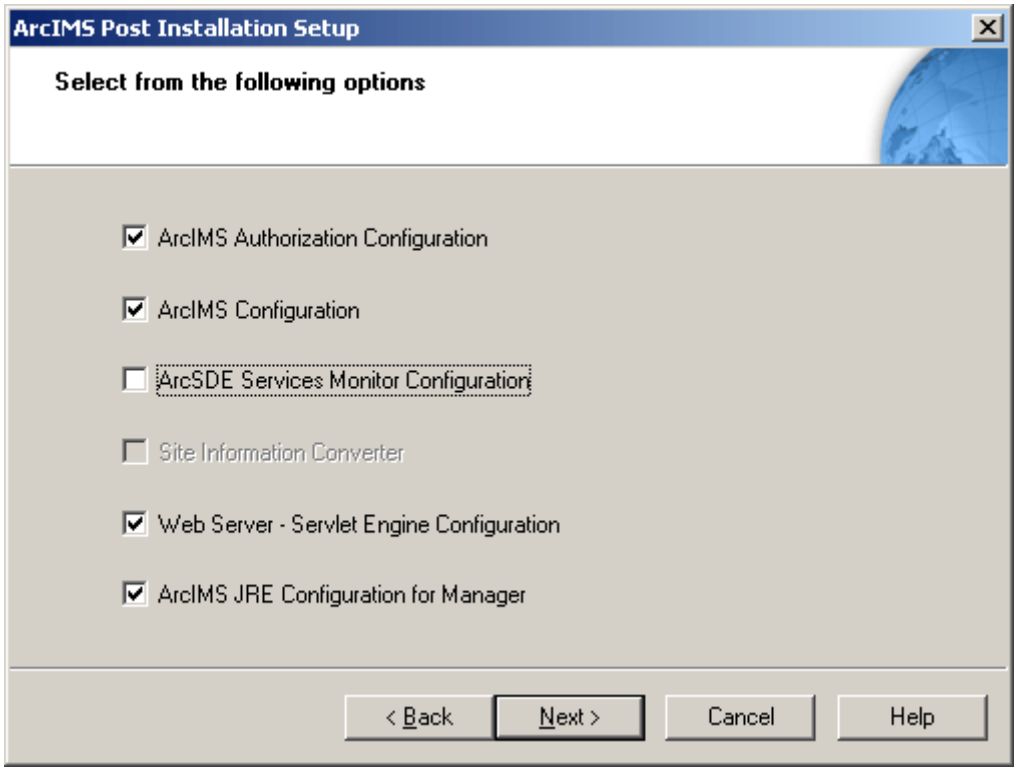

Figure 6. Screenshot of the ArcIMS Post-Installation Option Dialog Box.

The post install includes several steps to configure ArcIMS and to set up directories where ArcIMS will store important files. For more information on selecting options for the post install, refer to the ArcIMS installation guide at:

http://downloads.esri.com/support/documentation/ims_/ArcIMS_90_Windows_Install_Guide.pdf

Mostly, the defaults can be used. In the step to specify the Web server host name, enter the domain name of the server (ie. nowcoast.noaa.gov). This allows the *nowCOAST* viewer to correctly locate the images output by ArcIMS once installed. On the screen that asks for an account username and password under which to run ArcIMS, enter the username and password for the account that is being used for the install. As mentioned before, this should be a local account for the machine. The dialog will also ask the user to specify the directories where Apache and Tomcat are installed; these should be whichever directories were chosen in steps 2.3.1 and 2.3.2.

A.3. Apache, Tomcat Security, Performance Tuning, and Logging

Step 1. Copy ArcIMS components to the *nowCOAST* viewer webapp directory

Because the *nowCOAST* webapp is configured as the default in Tomcat, several ArcIMS components, which were installed during the ArcIMS install process in the Tomcat's ROOT webapp directory, must be copied to the *nowCOAST* webapp directory in order for ArcIMS to function correctly. To copy over the files, navigate to Tomcat's ROOT webapp directory, which should be located at:

```
<CATALINA_HOME>/webapps/ROOT/
```

From there, open the WEB-INF/classes subdirectories:

```
<CATALINA_HOME>/webapps/ROOT/WEB-INF/classes/
```

Within this directory, there should be four files as well as a directory called 'com'.

```
com/
Esrimap_prop
ServletConnector_Res.properties
ServletConnector_Res_en_US.properties
WMSEsrimap_prop
```

Copy all of these files and the /com directory to the corresponding directory within the Tomcat webapp (WEB-INF/classes). If the /classes subdirectory does not exist, create it.

```
C:/path/to/nowcoast/WEB-INF/classes/
```

These files comprise the ArcIMS servlet connector (a combination of Java classes and configuration files), and ArcIMS expects them to be located in Tomcat's default context. Once the files are copied over, this step is complete.

Step 2. Performance Tuning of the Java runtime environment for Tomcat

The Tomcat servlet connector is executed inside a Java virtual machine (JVM). When Tomcat is started as a Windows service, there are several options that may be passed to the JVM to tune aspects such as garbage collection, memory heap space, and others. The *nowCOAST* viewer requires tuning of the JVM in this way in order to prevent memory overflows and other problems from affecting how the viewer runs. Tomcat 5.5.x provides several methods by which to pass options to the JVM for tuning purposes. For Windows, there is a GUI interface that allows users to specify which options they would like to use. There is also an option to run a batch script from the command line to configure JVM options. Either is effective; however, this document will discuss using batch files to tune the Tomcat Windows service. Save the following two Windows batch files to Tomcat's /bin directory:

```
<CATALINA_HOME>/bin/
```

install.bat: (CATLINA_HOME and JAVA_HOME must be replaced with the file system locations of Tomcat and Java on the system – same as what was entered in the sytem environment variables during the ArcIMS install)

```
tomcat5 //IS//tomcat5 --DisplayName="Apache Tomcat 5" -Description="Apache
Tomcat 5.5.4 Server - http://jakarta.apache.org/tomcat/ " --
Install="<CATALINA_HOME>\bin\tomcat5.exe" --Startup=auto --Jvm="<JAVA_HOME>
\jre\bin\server\jvm.dll" –Classpath="<CATALINA_HOME>\bin\bootstrap.jar" --
StartMode=jvm --StopMode=jvm --StartClass=org.apache.catalina.startup.Bootstrap
--StartParams=start --StopClass=org.apache.catalina.startup.Bootstrap --
StopParams=stop --JvmOptions=-Xms4m#-Xmx256m#-XX:NewRatio=3#-
XX:MaxHeapFreeRatio=75#-XX:MinHeapFreeRatio=40#-
Xloggc:<CATALINA_HOME>\logs\gclog.txt#-XX:+PrintGCDetails#-
Dcatalina.home=<CATALINA_HOME>#-
Djava.endorsed.dirs=<CATALINA_HOME>\common\endorsed#-
Djava.io.tmpdir=<CATALINA_HOME>\temp --LogPath="<CATALINA_HOME>\logs" --
LogLevel=error --StdOutput="<CATALINA_HOME>\logs\stdout.log" --
StdError="<CATALINA_HOME>\logs\stderr.log"
```

delete.bat:

```
tomcat5 //DS//tomcat5
```

The delete.bat file will delete the Tomcat5 Windows service that was installed during the Tomcat install. Run delete.bat. This is necessary because a new Tomcat5 service must be created using the install.bat file. One of the options in the install.bat is '–JvmOptions'. This option tells the JVM how to configure memory usage for the Tomcat5 process and when/how to perform garbage collection, all factors that can influence performance. The *nowCOAST* viewer webapp can consume a large amount of memory, and this memory tuning is necessary to override the default memory configuration used by the JVM. Once the delete.bat file has completed processing, run install.bat to install a new Tomcat5 Windows service.

To verify that install.bat executed properly, open the Windows Service Manager under control panel and check that there is a service named 'Apache Tomcat 5" listed. More properties about the Tomcat5 service can be obtained by selecting 'Properties'. Additionally, in the same folder, in which the two batch files were placed, open the tomcat5w.exe file. This is a GUI based interface for editing the properties of the Tomcat Windows service. When opened it displays information about the Tomcat5 service similar to the properties window in the Windows Service Manager.

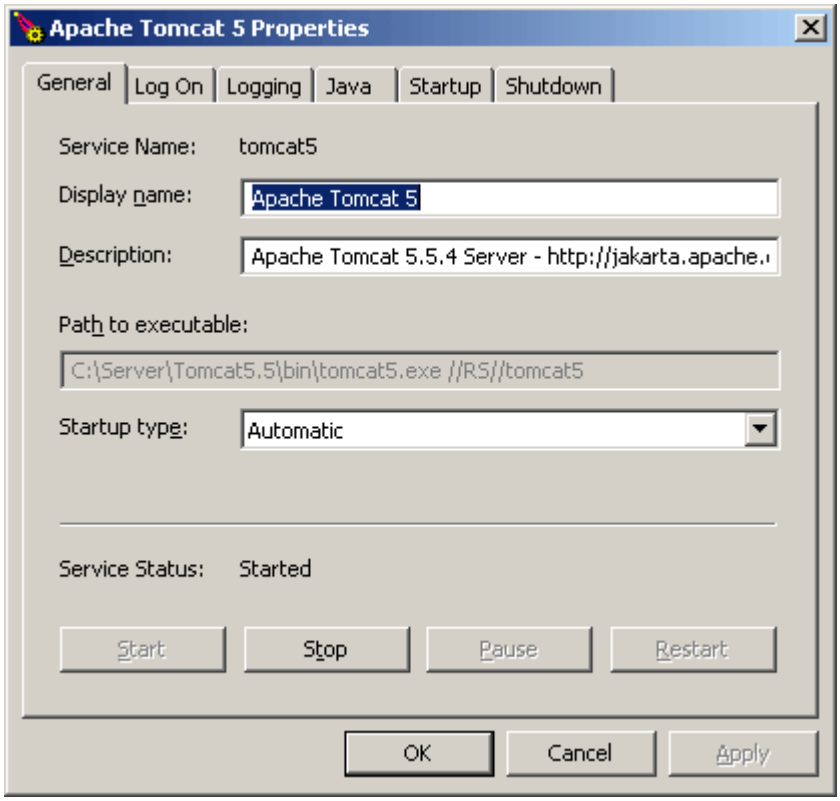

Figure 7. Screenshot of the Apache Tomcat 5 GUI-based Configuration Program.

Click on the Java tab in the window. Here the reader should see a list of the JVM tuning options that were in the install.bat file. Verify that the Xms and Xmx properties have the correct values (4m and 256m). These are the most critical tuning options to the *nowCOAST* viewer webapp to ensure that there is enough memory available in the Java heap space.

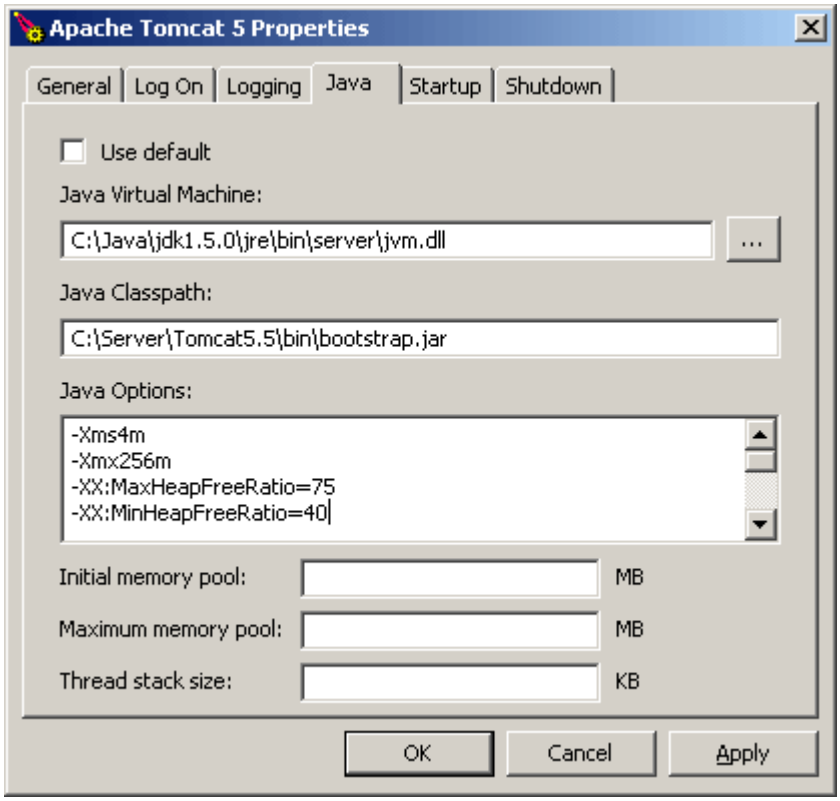

Figure 8. Screenshot of the Apache Tomcat 5 GUI-based Configuration Program Screenshot, Java Options.

If the parameters appear correct, there is no additional tuning necessary. The last step in configuring the Tomcat service is to set it up to restart if it fails. This is done through the Windows Services Manager. Open the properties window for the Tomcat5 service (as described above). On the Recovery tab, set each of the failure settings to 'Restart,' as in the graphic below:

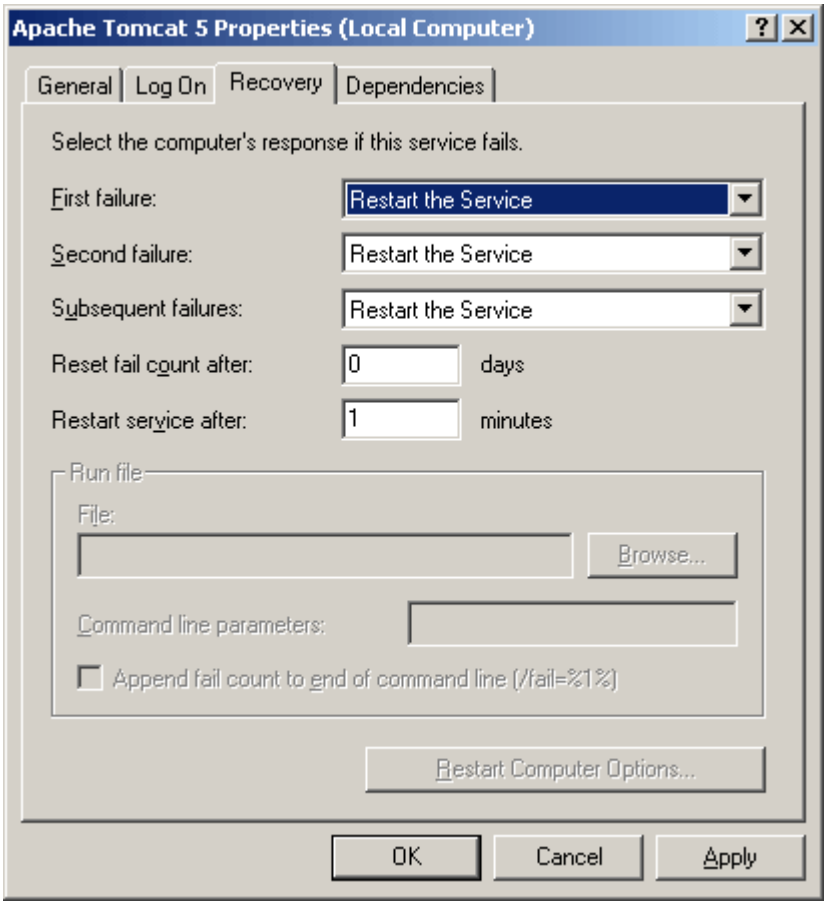

Figure 9. Screenshot of the Apache Tomcat 5 Windows Service Properties Dialog Box.

Tomcat should now be configured properly for *nowCOAST*. Additional references on Tomcat5 Windows service modification and JVM performance tuning is available at:

http://jakarta.apache.org/tomcat/tomcat-5.5-doc/windows-service-howto.html
http://java.sun.com/docs/hotspot/index.html
http://java.sun.com/j2se/1.4.2/docs/tooldocs/windows/java.html.

Step 3. Final Apache security measures

In addition to some security changes that were made to the Apache configuration following the installation, some other changes must be made after the ArcIMS post install has been run. This action is necessary, since the post install changes the Apache httpd.conf file. These changes modify some basic security settings, and additional changes may be required that are not documented here. Information about Apache security concerns can be found at:

http://httpd.apache.org/docs-2.0/misc/security_tips.html.

Open the Apache httpd.conf file and search for the section containing Alias directives that were inserted by ArcIMS. There should be in this section two Directory directives, one for the ArcIMS 'website' directory, and one for the 'output' directory. These two directories were specified during the ArcIMS post install (section 2.4.4). The default values are "C:/ArcIMS/Website" and "C:/ArcIMS/Ouput"; however if they were changed manually the paths will be different. The 'website' Directory directive should look like this:

```
<Directory "C:/ArcIMS/Website">
    Options Indexes MultiViews
    AllowOverride None
    Order allow,deny
    Allow from all
</Directory>
```

Change the directives within the Directory directive to the following:

```
<Directory "C:/ArcIMS/Website">
    Options None
    AllowOverride None
    Order allow,deny
    Deny from all
</Directory>
```

The 'output' Directory should look as follows, and may not need to have any changes made. This configuration allows clients access to the files within this directory but does not allow indexing of the directory. This is where ArcIMS is set up to place the images it outputs from a MapService.

```
<Directory "C:/ArcIMS/Output">
    Options MultiViews
    AllowOverride None
    Order allow,deny
    Allow from all
</Directory>
```

Step 4. Apache logfile rotation setup

The last step in the *nowCOAST* Apache configuration is setting up logfile rotation. *nowCOAST's* Apache Web server should be configured to auto-rotate log files on a monthly basis. The logfile rotation is accomplished by modifying the logfile directives included in the httpd.conf file.

The Apache install includes by default the `rotatelogs` utility (<Apache Root>/bin/rotatelogs). This is designed to provide log rotation functionality, however there are limitations to this program. *nowCOAST* is configured to use a separate utility called `cronolog` for this purpose. For more information about `cronolog` and to download the program visit http://cronolog.org.

nowCOAST uses the Windows distribution of cronolog v1.6.1, available at the same site. To install `cronolog`, download the files, unzip them, and place the cronolog.exe file in `<ApacheRoot>/bin` folder. Apache can then be configured to use `cronolog` in httpd.conf.

Setup in `http.conf`:

2. Find the ErrorLog directive. Comment out the existing ErrorLog directive and add the following directive:

   ErrorLog "|C:/Server/Apache2/bin/cronolog.exe C:/Server/Apache2/logs/nowcoast_error-%Y%m.log"
   (modify the directory path according to the directory that Apache is installed in – current nowCOAST configuration is C:/Server/Apache2)

3. Below the ErrorLog directive, find the CustomLog directives (i.e. 'access.log' 'referer.log' and 'agent.log'. Comment out all of these directives so that there are no CustomLogs specified. Below the last CustomLog directive, add the following two lines (modify directory paths accordingly):

   # Rotate logs using cronolog utility, http://cronolog.org/
   CustomLog "|C:/Server/Apache2/bin/cronolog.exe C:/Server/Apache2/logs/nowcoast_combined-%Y%m.log" combined

This CustomLog instructs Apache to pipe a CustomLog combined format log to the cronolog program. The output log is written to the /logs subdirectory in the format 'nowcoast_combined-YYYYMM.log'. This logfile is rotated at the end of each month. Also, the Apache error log is rotated every month and is named in a similar way: 'nowcoast_error-YYYYMM.log'. Both of these log files will be stored in this directory until removed manually.

Additional information about log file configuration using Apache directives can be found at:

http://httpd.apache.org/docs-2.0/logs.html.

More information about cronolog is available at:

http://cronolog.org/usage.html.

APPENDIX B. UPDATING *nowCOAST* DATABASE AND/OR MAP LAYERS

This section describes procedures for updating *nowCOAST* map layer shapefiles for point and polygon data. Currently, the procedure in Section B1 "Manual Shapefile Update" should be followed. Plans for automated shapefile creation using Feature Manipulation Engine in Section B2 will be established in the future. This procedure will also change when the Arc Spatial Database Engine (ArcSDE) is implemented on the *nowCOAST* server.

B1. Manual Shapefile Update

This section describes the steps to update *nowCOAST* shapefiles from the Microsoft Access database. This procedure is necessary only when updates are made to the database and is not performed on a regular schedule. Presently, the *nowCOAST* development team at CSDL will update the database, generate associated shapefiles (Steps 1, 2), and import shapefiles into the *nowCOAST* ArcIMS server (Step 3).

Step 1. Export MS Access Database Table

- 1.1 Choose table to export
- 1.2 Select Export, select file type `dBASE IV`
- 1.3 Save file in `exportfiles` directory as `filename`.dbf *

* `exportfiles` and `filename` will be determined by operators, `filename` is specific to a particular data layer.

Step 2. Convert Export File to Shapefile

- 2.1 Point Shapefile
  - 2.1.1 Using ArcView 3.2, import `filename`.dbf table to nowCOAST project (`nowCOAST.apr`)
  - 2.1.2 Select 'Add Event Theme', select `filename`.dbf in 'Table' pulldown, specify Longitude and Latitude fields for 'X field' and 'Y field' pulldowns, respectively
  - 2.1.3 Select 'Convert to Shapefile', specify `filename`.shp, save in `shapefiles` directory *
- 2.2 Polygon Shapefile: Associate `filename`.dbf with appropriate shapefile
  - 2.2.1 If only attribute data have changed (there is no new shapefile information), copy the .shp and .shx files from the old shapefile and rename according to `filename`.dbf. Open this new shapefile (`filename`.shp) in ArcView 3.2 in the `nowCOAST.apr` project to verify its accuracy
  - 2.2.2 If there is new shapefile information (polygonal boundaries have changed), rename the new .shp and .shx files according to `filename`.dbf. Open this new shapefile (`filename`.shp) in ArcView 3.2 in the `nowCOAST.apr` project to verify its accuracy
  - 2.2.3 In either case, save `filename`.shp in the `shapefiles` directory *

* `shapefiles` will be determined by operators.

Step 3. Import Shapefile to ArcIMS

3.1 Copy `filename`.shp to *nowCOAST* MapService's `data` directory *

3.2 Edit *nowCOAST*'s map configuration file (.axl file) according to new shapefile

3.2.1 Change 'name' attribute of '<DATASET>' tag of corresponding '<LAYER>' tag (substitute name of shapefile being replaced with `filename`.shp).

3.2.2 Save map configuration file as `nowCOASTyymmdd`.axl where `yymmdd` equals current date.

3.3 Update *nowCOAST* MapService in ArcIMS Administrator

3.3.1 Open 'Properties' dialog of *nowCOAST* MapService

3.3.2 Change filename of nowCOASTxxxxxx.axl file in 'Map File' field to `nowCOASTyymmdd`.axl as in above

3.3.3 Hit 'Refresh Service' button

3.3.4 Save changes and exit ArcIMS Administrator

3.4 Verify changes to *nowCOAST*

3.4.1 Open *nowCOAST* in web browser

3.4.2 Verify proper function of *nowCOAST*

3.4.3 Verify new shapefile is displayed in *nowCOAST*

* `data` will depend upon the server's ArcIMS installation.

Table 4 is a list of the GIS Layers and corresponding shapefiles in *nowCOAST*. These files are located in C:\ArcIMS\data\Shapefiles.

Table 4. GIS Layers and corresponding names of shapefiles in *nowCOAST*.

Layer	File Name (.shp)
Observations:	
Meteorological	obs_pt_wx_YYYYMMDD
Oceanographic	obs_pt_oc_YYYYMMDD
River	obs_pt_riv_YYYYMMDD
Water Quality	obs_pt_wq_YYYYMMDD
Weather Radars	obs_pt_wr_YYYYMMDD
Boundary Layer Profiles	obs_pt_blp_YYYYMMDD
Upper Air Soundings	obs_pt_rs_YYYYMMDD
HF Radar Surface Currents	obs_pt_hfr_YYYYMMDD
Web Cameras	obs_pt_wc_YYYYMMDD
Predictions:	
Astronomical Tidal	pred_pt_at_YYYYMMDD
Forecast Model Guidance:	
Meteorological	fcst_pt_wx_YYYYMMDD
Oceanographic	fcst_pt_oc_YYYYMMDD
River	fcst_pt_riv_YYYYMMDD
Surface Forecasts:	
Surface Weather (WFO Zones)	fcst_pg_cfz_YYYYMMDD
Surf Zone	fcst_pg_sz_YYYYMMDD
Coastal and Offshore	fcst_pg_coz_YYYYMMDD
High Seas	fcst_pg_hsz_YYYYMMDD
Map Background:	
Exclusive Economic Zones	us_eez_boundary
US States - General	states
World Lakes	lakes
World Rivers	rivers
World Countries	cntry02
World Countries Labels	cntry02
Watersheds	estuarine-wsds
Coastal Labels	coastal_labels
US Cities	us_cities5000up
US Interstates	us_interstates
Major Roads	us_roads_na
Lakes	us_lakes_2005
Rivers	us_rf1_primary_display
Urban Areas	urban_areas
US/Canada Background	us_canada
Florida Keys NMS	fknms_py

The shapefiles are located in C:\ArcIMS\data\Shapefiles.

B2. Feature Manipulation Engine (FME) Shapefile Updating

Once procedures have been established, shapefile updates necessitated by changes in the MS Access database will be automated using FME. This process will require quality control steps/logfile monitoring by NOS personnel to ensure successful inclusion of automated shapefiles into *nowCOAST*.

APPENDIX C. CONFIGURATION FILES FOR *nowCOAST* SERVER

This section contains text versions of configuration files used in the *nowCOAST* server configuration (Appendix A). The two files are called mod_jk.conf and workers.properties, and they are used to configure the mod_jk connector. AJP13 protocol allows the Apache HTTP server to work with Tomcat Servlet engine in the *nowCOAST* server. Explanations about the use of each of these files are given in Appendix A. Consult Appendix A for instructions on using these files for the *nowCOAST* server installation.

mod_jk.conf:

```
<IfModule !mod_jk.c>
  LoadModule jk_module modules/mod_jk.dll
</IfModule>

JkWorkersFile "C:\Server\Tomcat5.5\conf\workers.properties"
JkLogFile "C:\Server\Tomcat5.5\logs\mod_jk.log"

JkLogLevel error




<VirtualHost *:80 >
    ServerName nowcoast.noaa.gov
     DocumentRoot "C:/Shared/nowcoast_jspbeta/build"


# Code to display the server update page in Apache when Tomcat is down:
#   change DocumentRoot above to C:/Shared/nowcoast_jspbeta and uncomment the
#   directive below and nowcoast.noaa.gov will default to server_update.html
#                       <Directory "C:/Shared/nowcoast_jspbeta">
#                                       Options -Indexes
#                                       AllowOverride None
#                                       Order allow,deny
#                                       Allow from all
#                                       DirectoryIndex server_update.html
#                       </Directory>


# Code for redirect from /nowcoast_jspbeta

                        Alias /nowcoast_jspbeta "C:/Shared/nowcoast_jspbeta/redirect"
                        <Directory "C:/Shared/nowcoast_jspbeta/redirect">
                                        Options -Indexes
                                        AllowOverride None
                                        Order allow,deny
                                        Allow from all
                                        DirectoryIndex redirect.html
                        </Directory>


# Configure Apache for Version 2 viewer setup:includes, indexing:
                        <Directory "C:/Shared/nowcoast_jspbeta/build">
                                        Options -Indexes
                                        AllowOverride None
                                        Order allow,deny
                                        Allow from all
                                        #SetOutputFilter INCLUDES
                        </Directory>
                        <Directory "C:/Shared/nowcoast_jspbeta/build/help">
                                        Options +IncludesNOEXEC
                                        AllowOverride None
                                        Order allow,deny
                                        Allow from all
                                        SetOutputFilter INCLUDES
```

```
                                DirectoryIndex infodepot.shtml
        </Directory>



   #Serve all .jsp's from Tomcat with this JKMount directive

   JkMount /*.jsp ajp13


   JkMount /servlet ajp13
   JkMount /servlet/* ajp13

   JkMount /balancer ajp13
   JkMount /balancer/* ajp13

   JkMount /esriadmin ajp13
   JkMount /esriadmin/* ajp13

   JkMount /jsp-examples ajp13
   JkMount /jsp-examples/* ajp13

   JkMount /servlets-examples ajp13
   JkMount /servlets-examples/* ajp13

   JkMount /webdav ajp13
   JkMount /webdav/* ajp13

   JkMount /tomcat-docs ajp13
   JkMount /tomcat-docs/* ajp13

   JkMount /manager ajp13
   JkMount /manager/* ajp13



</VirtualHost>




# Configuration for nowcoast.ncd.noaa.gov

#  - This configuration is set up so Apache serves all static HTML and images and Tomcat
#                                is responsible for the .jsp files only - better division of labor and allows for
#                                use of Apache <Directory> tags to configure SSIs and Indexing (theoretically)....
#
<VirtualHost *:80 >
        ServerName nowcoast.ncd.noaa.gov
        DocumentRoot "C:/Shared/nowcoast_jspbeta/build"


# putting the default / Directory within this Virtualhost means that it will only map to directories
#   within the DocumentRoot above (ie it doesn't replace the same default / Directory in httpd.conf
#   - Changed to C:/Shared/nowcoast_jspbeta/build here instead
#
#<Directory />
<Directory "C:/Shared/nowcoast_jspbeta/build">
        Options -Indexes
        AllowOverride None
```

```
                    Order allow,deny
                    Allow from all
                    #SetOutputFilter INCLUDES
</Directory>

# this adds the ability for SSIs to the /help directory
#                   - also, the DirectoryIndex specifies the file to display by default on a "help/" request with no specific filename
<Directory "C:/Shared/nowcoast_jspbeta/build/help">
                    Options +IncludesNOEXEC
                    AllowOverride None
                    Order allow,deny
                    Allow from all
                    SetOutputFilter INCLUDES
                    DirectoryIndex intro.shtml
</Directory>




                    # Configuration with the Version 2 viewer webapp serving as the default web app in Tomcat:
                    # *.jsp ensures that Tomcat serves the .jsp pages without a specific webapp path, whereas all
                    #                   other resuources without a specific webapp path will be served via Apache from the localhost
root
                    #                   which also is the root of the Version 2 viewer webapp in Tomcat.
("C:/Shared/nowcoast_jspbeta/build")

                    JkMount /*.jsp ajp13




</VirtualHost>
```

workers.properties:

```
# workers.properties -
#
# This file provides jk derived plugins with the needed information to
# connect to the different tomcat workers.  Note that the distributed
# version of this file requires modification before it is usable by a
# plugin.
#
# As a general note, the characters $( and ) are used internally to define
# macros. Do not use them in your own configuration!!!
#
# Whenever you see a set of lines such as:
# x=value
# y=$(x)\something
#
# the final value for y will be value\something
#
# Normaly all you will need to do is un-comment and modify the first three
# properties, i.e. workers.tomcat_home, workers.java_home and ps.
# Most of the configuration is derived from these.
#
# When you are done updating workers.tomcat_home, workers.java_home and ps
# you should have 3 workers configured:
#
# - An ajp12 worker that connects to localhost:8007
# - An ajp13 worker that connects to localhost:8009
# - A jni inprocess worker.
# - A load balancer worker
#
# However by default the plugins will only use the ajp12 worker. To have
# the plugins use other workers you should modify the worker.list property.
#
#

# OPTIONS ( very important for jni mode )

#
# workers.tomcat_home should point to the location where you
# installed tomcat. This is where you have your conf, webapps and lib
# directories.
#
workers.tomcat_home=C:\Server\Tomcat5.5

#
# workers.java_home should point to your Java installation. Normally
# you should have a bin and lib directories beneath it.
#
workers.java_home=C:\Java\jdk1.5.0

#
# You should configure your environment slash... ps=\ on NT and / on UNIX
# and maybe something different elsewhere.
#
ps=/

#
#------ ADVANCED MODE ----------------------------------------------
#-------------------------------------------------------------------
```

```
#

#
#------ DEFAULT worket list ----------------------------------------
#----------------------------------------------------------------------
#
#
# The workers that your plugins should create and work with
#
# Add 'inprocess' if you want JNI connector
worker.list=ajp12, ajp13
# , inprocess


#
#------ DEFAULT ajp12 WORKER DEFINITION ---------------------------
#----------------------------------------------------------------------
#

#
# Defining a worker named ajp12 and of type ajp12
# Note that the name and the type do not have to match.
#
worker.ajp12.port=8007
worker.ajp12.host=localhost
worker.ajp12.type=ajp12
#
# Specifies the load balance factor when used with
# a load balancing worker.
# Note:
#  ----> lbfactor must be > 0
#  ----> Low lbfactor means less work done by the worker.
worker.ajp12.lbfactor=1

#
#------ DEFAULT ajp13 WORKER DEFINITION ---------------------------
#----------------------------------------------------------------------
#

#
# Defining a worker named ajp13 and of type ajp13
# Note that the name and the type do not have to match.
#
worker.ajp13.port=8009
worker.ajp13.host=localhost
worker.ajp13.type=ajp13
#
# Specifies the load balance factor when used with
# a load balancing worker.
# Note:
#  ----> lbfactor must be > 0
#  ----> Low lbfactor means less work done by the worker.
worker.ajp13.lbfactor=1

#
# Specify the size of the open connection cache.
#worker.ajp13.cachesize

#
#------ DEFAULT LOAD BALANCER WORKER DEFINITION ----------------------
```

```
#---------------------------------------------------------------------
#

#
# The loadbalancer (type lb) workers perform wighted round-robin
# load balancing with sticky sessions.
# Note:
#  ----> If a worker dies, the load balancer will check its state
#        once in a while. Until then all work is redirected to peer
#        workers.
worker.loadbalancer.type=lb
worker.loadbalancer.balanced_workers=ajp12, ajp13


#
#------ DEFAULT JNI WORKER DEFINITION--------------------------------
#---------------------------------------------------------------------
#

#
# Defining a worker named inprocess and of type jni
# Note that the name and the type do not have to match.
#
worker.inprocess.type=jni

#
#------ CLASSPATH DEFINITION ---------------------------------------
#---------------------------------------------------------------------
#

#
# Additional class path components.
#
worker.inprocess.class_path=$(workers.tomcat_home)$(ps)lib$(ps)tomcat.jar

#
# Setting the command line for tomcat.
# Note: The cmd_line string may not contain spaces.
#
worker.inprocess.cmd_line=start

# Not needed, but can be customized.
#worker.inprocess.cmd_line=-config
#worker.inprocess.cmd_line=$(workers.tomcat_home)$(ps)conf$(ps)server.xml
#worker.inprocess.cmd_line=-home
#worker.inprocess.cmd_line=$(workers.tomcat_home)

#
# The JVM that we are about to use
#
# This is for Java2
#
# Windows
worker.inprocess.jvm_lib=$(workers.java_home)$(ps)jre$(ps)bin$(ps)classic$(ps)jvm.dll
# IBM JDK1.3
#worker.inprocess.jvm_lib=$(workers.java_home)$(ps)jre$(ps)bin$(ps)classic$(ps)libjvm.so
# Unix - Sun VM or blackdown
#worker.inprocess.jvm_lib=$(workers.java_home)$(ps)jre$(ps)lib$(ps)i386$(ps)classic$(ps)libjvm.so

#
```

```
# And this is for jdk1.1.X
#
#worker.inprocess.jvm_lib=$(workers.java_home)$(ps)bin$(ps)javai.dll



#
# Setting the place for the stdout and stderr of tomcat
#
worker.inprocess.stdout=$(workers.tomcat_home)$(ps)logs$(ps)inprocess.stdout
worker.inprocess.stderr=$(workers.tomcat_home)$(ps)logs$(ps)inprocess.stderr

#
# Setting the tomcat.home Java property
#
#worker.inprocess.sysprops=tomcat.home=$(workers.tomcat_home)

#
# Java system properties
#
# worker.inprocess.sysprops=java.compiler=NONE
# worker.inprocess.sysprops=myprop=mypropvalue

#
# Additional path components.
#
# worker.inprocess.ld_path=d:$(ps)SQLLIB$(ps)bin
```