Using of CORBA/Java technologies for accessing of environmental databases via Internet in framework of the Black Sea Web project

H. Berle[1], V. Miroshnichenko[2], V. Lyubartsev[2], I. Kalinkin[3]

1. TERMA Elektronik A/S (TERMA), Bregneroedvej 144, DK-3460 Birkeroed, Denmark
2. Marine Hydrophysical Institute (MHI), 2 Kapitanskaya St., Sevastopol 335000, Ukraine
3. Moscow State University (MSU), Vorobevi Gori, RU-119899, Moscow, Russia

Introduction

Assessment of ecological situation in specific geographical region requires of analysis of environmental information collected all over the region. The good idea is to gather all collected data from different sources into one database and use it as a main data source, but historically ecological information is accumulated in different organisations and different countries and stored in various databases in different formats. Internet communications open wide possibilities for information exchange, but don't solve automatically problem with access to data stored in different organisations.

INCO COPERNICUS Project **BLACK SEA WEB** (project number 977005) is aimed to develop a Demonstrator for a Black Sea Marine Environmental Management Support System, which provides access into each others data of organisations dealing with marine environmental data (which essentially are ecological data) of the Black Sea, on an independent base and maintaining the control of their own data. The main idea is to develop a uniform query facility for searching on multiple oceanographic database catalogues via Internet, where these databases are geographically distributed and heterogeneous. This concept results in several connected databases, which are physically separated, but appear as one. Each of the databases has its own query interface, which may or may not be available from the World Wide Web. Innovative ***CORBA/Java*** technologies are chosen for solving of task to provide data access in heterogeneous database via Internet.

Definitions, acronyms and abbreviations used in article

	Definition
CGC	Central Guidance Catalogue
CGCS	Central Guidance Catalogue Server
CGCSO	Central Guidance Catalogue Server Object
GC	Guidance Catalogue
GCS	Guidance Catalogue Server
CORBA	Common Object Request Broker Architecture
HTML	Hyper Text Markup Language
HTTP	Hyper Text Transfer Protocol
IDL	Interface Definition Language
IIOP	Internet inter-ORB Protocol
Java	Machine independent programming language
LGC	Local Guidance Catalogue
LGCS	Local Guidance Catalogue Server
LGCSO	Local Guidance Catalogue Server Object
LQS	Local Query Server
LQSO	Local Query Server Object
LS	Local Server
LSDB	Local Scientific Database
MC	Mapping Catalogue
ORB	Object Request Broker
RDBMS	Relational Database Management System
SC	Scientific Client
SCJA	Scientific Client Java Applet
UQ	Uniform Query
WWW	World Wide Web

Architectural Model

The Common Object Request Broker Architecture (CORBA) provides a common environment where distributed objects can communicate, operate on different hardware and software platforms and work with a variety of programming languages. It is defined by the Object Management Group (OMG), a consortium of over 800 organisations across the IT industry. The essential concept in CORBA is the Object Request Broker (ORB) - middleware (open software bus) that establishes the client-server relationships between objects. It allows CORBA objects to invoke one another without knowing where the objects they access reside or in what language the requested objects are implemented. The Interface Definition Language (IDL) is used to define the interfaces to CORBA objects. The Internet Inter-ORB Protocol (IIOP) specifies how object messages are exchanged using TCP/IP connections, i.e. IIOP specifies a standardised interoperability protocol for the Internet.

Since Java™ IDL - an Object Request Broker from Java™ - is provided with the JDK 1.2, it adds CORBA capability to the Java™ platform. Java IDL enables distributed Web-enabled Java technology applications to transparently invoke operations on remote network services using the industry standard OMG IDL and IIOP. CORBA client objects can be embedded into Java Applet and invoked by user via Internet all over the World by means of standard WWW Browser.

All described above features make CORBA in conjunction with Java a very powerful tool for developing of comprehensive client-server applications. So the ***CORBA/Java*** technology was chosen in **BLACK SEA WEB** project for constructing of systems for accessing of distributed heterogeneous environmental databases via Internet. System is now under constructions in framework of. It contains next main CORBA objects:

- Scientific Client Java Applet (SCJA);
- Central Guidance Catalogue Server Object (CGCSO);
- Local Guidance Catalogue Server Object (LGCSO);
- Local Query Server Object (LQSO).

The **Figure 1** constitutes the top-level view of the BlackSeaWeb architecture.

The System allows Scientific User to formulate a query using standard Web Browser or an applet viewer. The user can select query parameters via the user interface. The user interface contains text fields and lists of values, which the user can use to formulate the uniform query. Furthermore the user is able to select an area of interest, from a map of the region.

The information needed to fill the user interface is fetched from the Guidance Catalogue Server (GCS), which holds information about the fields and field values which the different Local Query Servers (LQS) is able to handle.

When the user has finished the query formulation, he submits the uniform query to the LQSs. The LQSs will convert the Uniform Query (UQ) to a local query (LQ) using a local Mapping Catalogue (MC). The MC describes how the uniform query fields can be converted to the locally used fields. The LQ is then executed on the Local Scientific Database (LSDB). The result fields of the LQ must then be converted to uniform fields. This is done using the MC.

The LQS then returns the result to the user. In first instance the result contains a metadata description of the datasets which can be requested later.

The user can then browse through the metadata results from the different local query servers to identify the dataset(s) which he is interested in. These datasets can then be selected and fetched from the local query server, which holds the datasets.

The approach described above establishes the GCS as the main system component providing almost all interactions between the Scientific Client and LQSs. At the present time the Black Sea region countries do not have reliable Internet connections. Therefore, architecture with only one GCS seems to be slow and unreliable in this situation. A possible solution of this problem is to create copies of the GCS, so-called "mirrors", on each LQS host. In this case the client may enter into the system from the nearest GCS. Moreover, if the client needs only data located on this LQS it is not necessary at all to connect other LQS.

It should be noted that the need for creating GCS "mirrors" at LQSs does not change anything in the Client-GCS-LQS interactions. The "mirroring functionality" is an independent top level task and is described separately in section *Guidance Catalogues Maintenance*.

Some of the important scenarios in the system, namely the *Data Retrieving* and *Guidance Catalogue Maintenance* scenarios are described below for more distinct describing of system functioning.

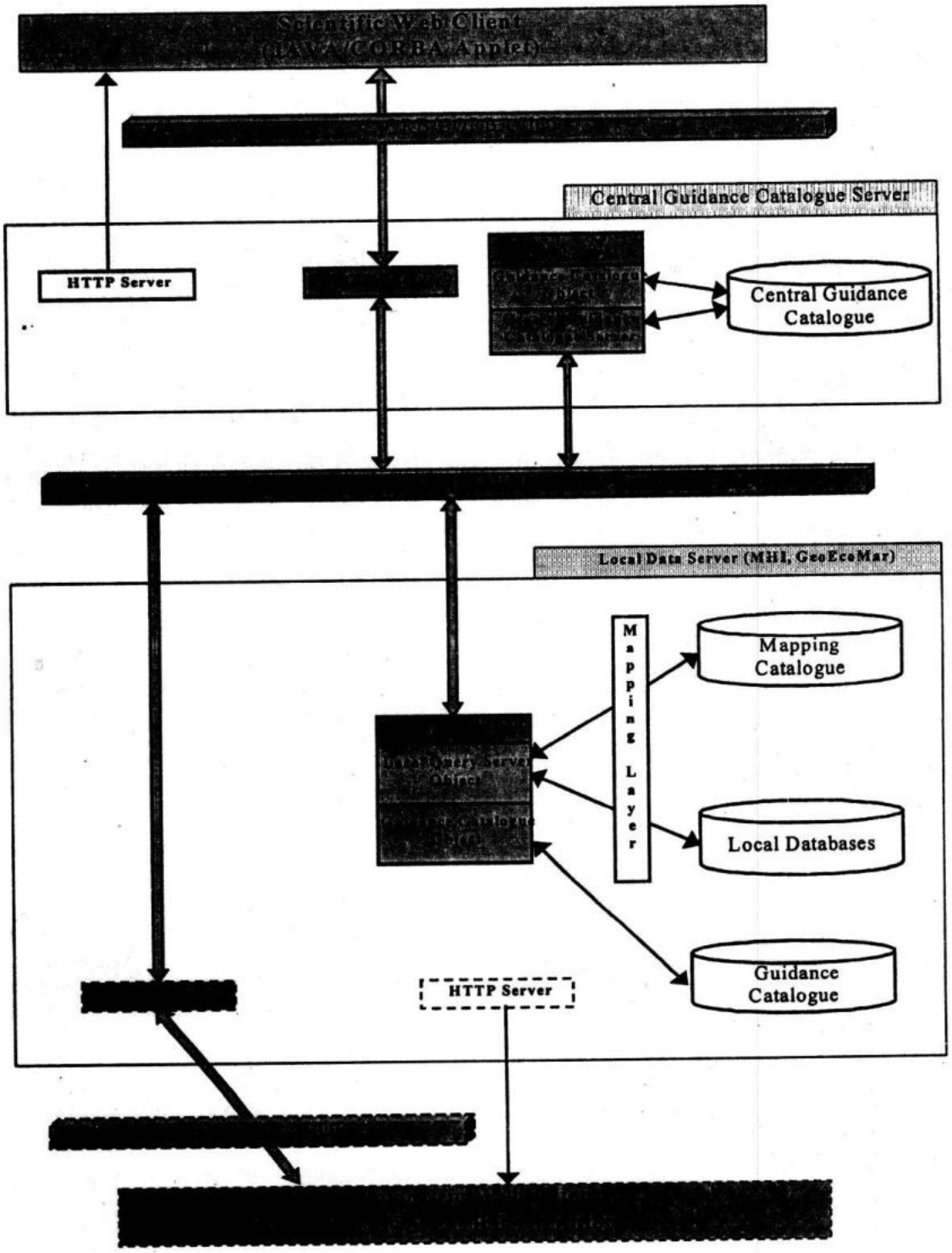

Figure 1. Architectural Model: general view

Data Retrieving Scenario

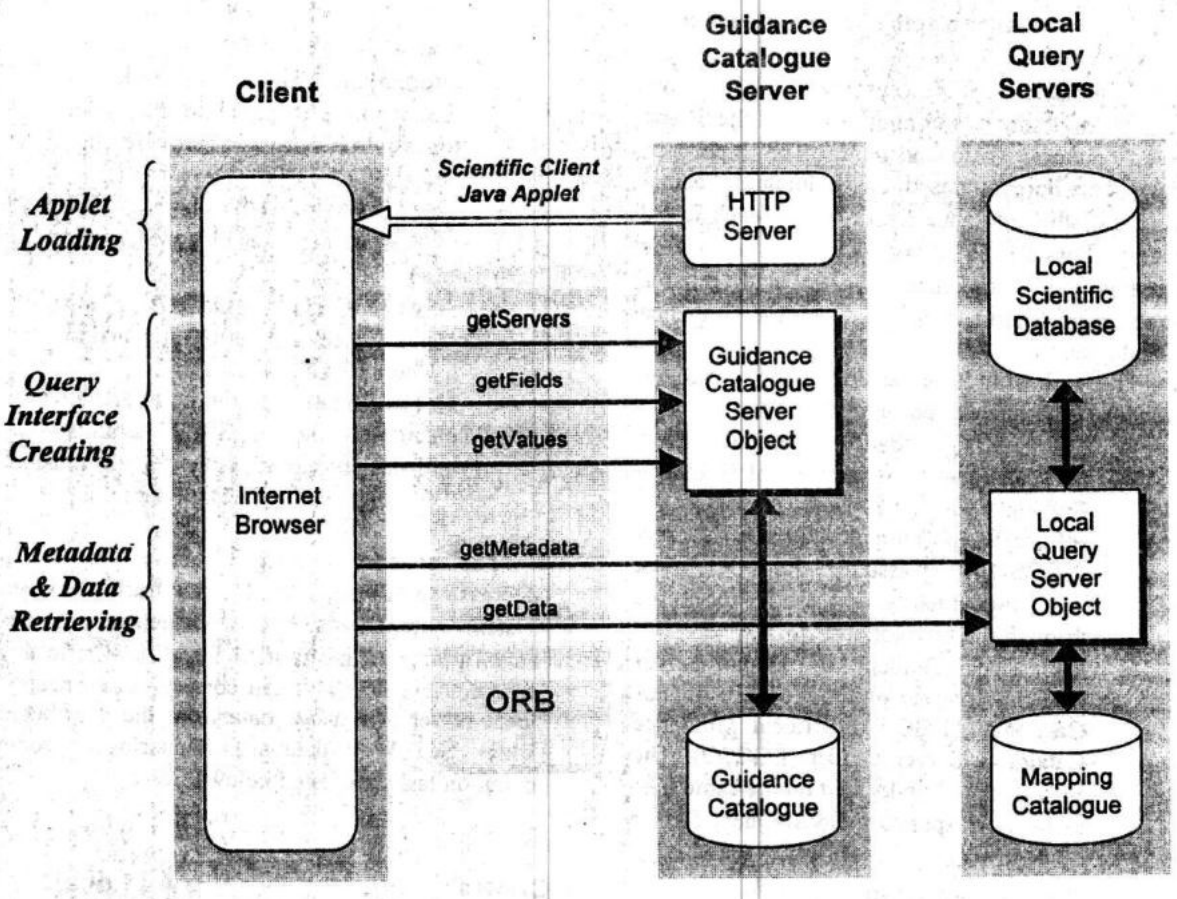

Figure 2. Data Retrieving Scenario

1. The User opens an HTML page with the Scientific Client Java applet in a standard HTML browser (Internet Explorer or Netscape).
2. The Browser starts the Java applet to create the Query Interface.
3. The applet asks the CORBA server for the reference to the Guidance Catalogue Server Object.
4. The applet receives the reference to the Guidance Catalogue Server Object.
5. The applet asks the Guidance Catalogue Server Object for Query parameters, using its getServers, getFields and getValues methods.
6. The applet creates the Uniform Query Interface.
7. The User uses the Uniform Query Interface to formulate a Uniform Query.
8. The applet obtains a list of enabled Local Query Server (LQS) Objects from the CORBA server.
9. The applet invokes the method getMetadata for each LQS, passing the Uniform Query as method parameters.
10. The LQS translates the Uniform Query into a local database query using the Mapping Catalogue.
11. The LQS executes the query and returns found metadata (list of stations, profiles, etc) to the applet.
12. The applet combines responses from different LQS, displays metadata in the table and draws station position marks on the Black Sea Map.
13. The User selects station(s) and pushes the Fetch data button.
14. The applet invokes the method getData for each LQS, passing a MetadataID as a parameter.
15. The LQS translates the query into the local database language using the Mapping Catalogue.
16. The LQS executes the local query and returns the result to the applet.
17. The applet combines results from different LQS and displays a table with data found, in a new HTML Browser window. The User has the possibility to view, print, and copy to clipboard or save the table into a local file.

Guidance Catalogues Maintenance

As mentioned, it is necessary to take the low quality of Internet communication in the Black Sea countries into account. The approach suggested here is based on maintaining actual copies of the Guidance Catalogue on each Local Query Server. To carry out this task it is necessary to implement a fast and reliable mechanism to update and replicate the Guidance Catalogue.

The following architecture, using catalogue version technique can be suggested:

- The Central Guidance Catalogue Server (CGCS) consists of the Central Guidance Catalogue (CGC) and the Central Guidance Catalogue Server Object (CGCSO). It is assumed the CGCS must work permanently and easily accessible through the Internet.
- Each Local Guidance Catalogue Server (LGCS) consists of the Local Guidance Catalogue (LGC) and Local Guidance Catalogue Server Object (LGCSO). The LGCS should be located on the same host as the corresponding Local Query Server (LQS). It is assumed that several LGCSs are linked to the **Black Sea Web**.
- Both CGC and LGCs keep their version information. The main task of GC replication is to maintain the actual state for all LGCs, i.e. to provide the same version for all CGs.
- The connection between the CGCS and LGCSs is established in the following cases:
- Some LGCS/LQS starts after a failure or pause, checks the CGC version and update its LGC if necessary.
- The information in the particular LGC is changed by the LGCS/LQS administrator, and as a consequence the CGC is updated. The CGC then updates all the LGCs.

The scheme described above allows to decrease significantly the Internet traffic between system components but nevertheless provides actual guidance information for all LGCs. In this case a scientific client can connect to the nearest local server to retrieve data from the distributed **Black Sea Web** database. Scenarios or some common tasks are listed below.

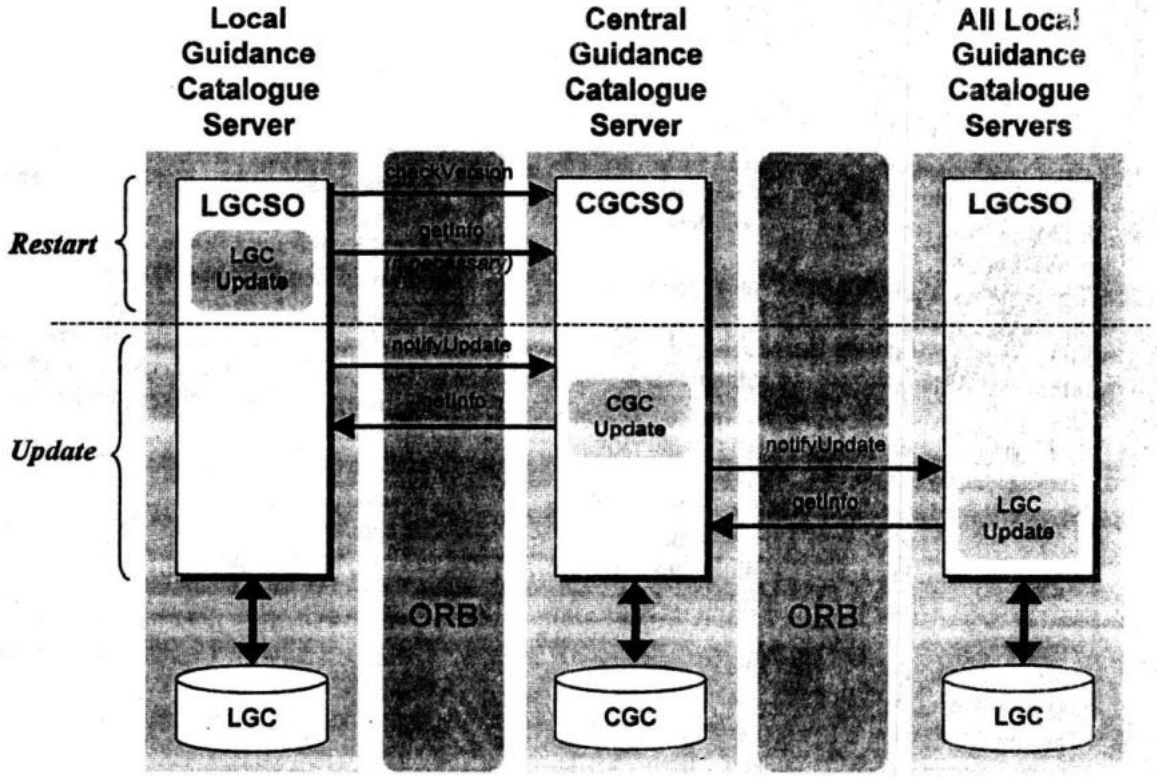

Figure 3. Guidance Catalogue Maintenance

Local Server Restart Scenario

1. The LGCSO asks the CORBA server for the CGCSO reference.
2. The LGCSO receives the reference to the CGCSO.
3. The LGCSO invokes the GetVersion method of the CGCSO.
4. If the CGC version differs from the LGC version then the LGCSO calls the GetInfo method of the CGCSO to retrieve the latest information and update the LGC.

Guidance Catalogues Update Scenario

1. The local server administrator updates the part of the LGC describing the corresponding LQS.
2. The LGCSO asks the CORBA server for the CGCSO reference.
3. The LGCSO receives the reference to the CGCSO.
4. The LGCSO invokes the NotifyUpdate method of the SGCSO.
5. The SGCSO invokes the GetInfo method of this LGCSO to retrieve the updated part of LGC.
6. The part of the CGC describing the updated local server is replaced with the retrieved information.
7. The CGC version is incremented.
8. The CGCSO invokes the NotifyUpdate method for all LGCSOs in the system.
9. Each LGCS invokes the GetInfo method of the CGCS to retrieve the updated CGC.
10. The LGC is replaced with the retrieved CGC.
11. The LGC version corresponds to the CGC version.

Note: getInfo is a short hand for the combination of the getServers, getFields and getValues methods.

Description of the architectural components

The Black Sea Web User Interface

User Interface is implemented in Scientific Client Java Applet (SCJA) which can be invoked in standard WWW Browser. WWW browser, which executes SCIA, becomes CORBA client, which can interact with CORBA servers – LGCSO, LQSO – provided access to distributed database.

User Interface consists of next pages:

- Search Criteria Page;
- Spatial Coverage page;
- Query Result page.

The Search Criteria page enables the user to build a query, which is a combination of the following fields:

- Starting Date and Time;
- Ending Date and Time;
- Organisation Name;
- Ship Name;
- Parameter Name;
- Minimum and Maximum Depth;

The information used to build the above page is retrieved from the Guidance Catalogue Server. Page contains also Data Centres checkboxes, which enables the user to select data centre(s) he wants to include in the search. When a checkbox is ticked the query is send to the corresponding data centre, i.e. to the correspondent CORBA server functioning on this data centre. At least one checkbox must be ticked, before a query can be made. The default is that all data centres are included in the search.

The Spatial Coverage page (see **Figure 4.**) allows the user to select a region of interest. The region of interest is used to establish the geographic boundaries for the Uniform Query. The main object on this page is the Map of the Black Sea region. The Map is developed in MSU and as a matter of fact is implementations of the Black Sea GIS in Java Bean. It can be Zoomed and Unzoomed, different layers –Rivers, Political Boundaries – can be displayed on the map.

The Map is used to select a geographical region of interest. The user can select an area of interest by clicking on the image and while holding the mouse button (s)he can drag the cursor to draw a rectangle on the map. While the rectangle is being drawn the corresponding latitude and longitude are updated in the four fields Min. Latitude, Min Longitude, Max. Latitude and Max. Longitude.

Page contains Submit Query and Reset buttons. The Submit Query button is used to submit a query after the user has filled in the query parameters and selected the region of interest. It initiates CORBA client to invoke methods of correspondent CORBA servers for obtaining of query result. The Reset button is used to reset the current selected query parameters to the default values.

The Query Result page displays the result of a Query. The page contains a Map with red crosses for each found station or profile with requested environmental data. The metadata information about each station or profile can be obtained by moving the cursor over one of the red crosses or by clicking one of the red crosses. The result is then displayed in the text boxes below the Map.

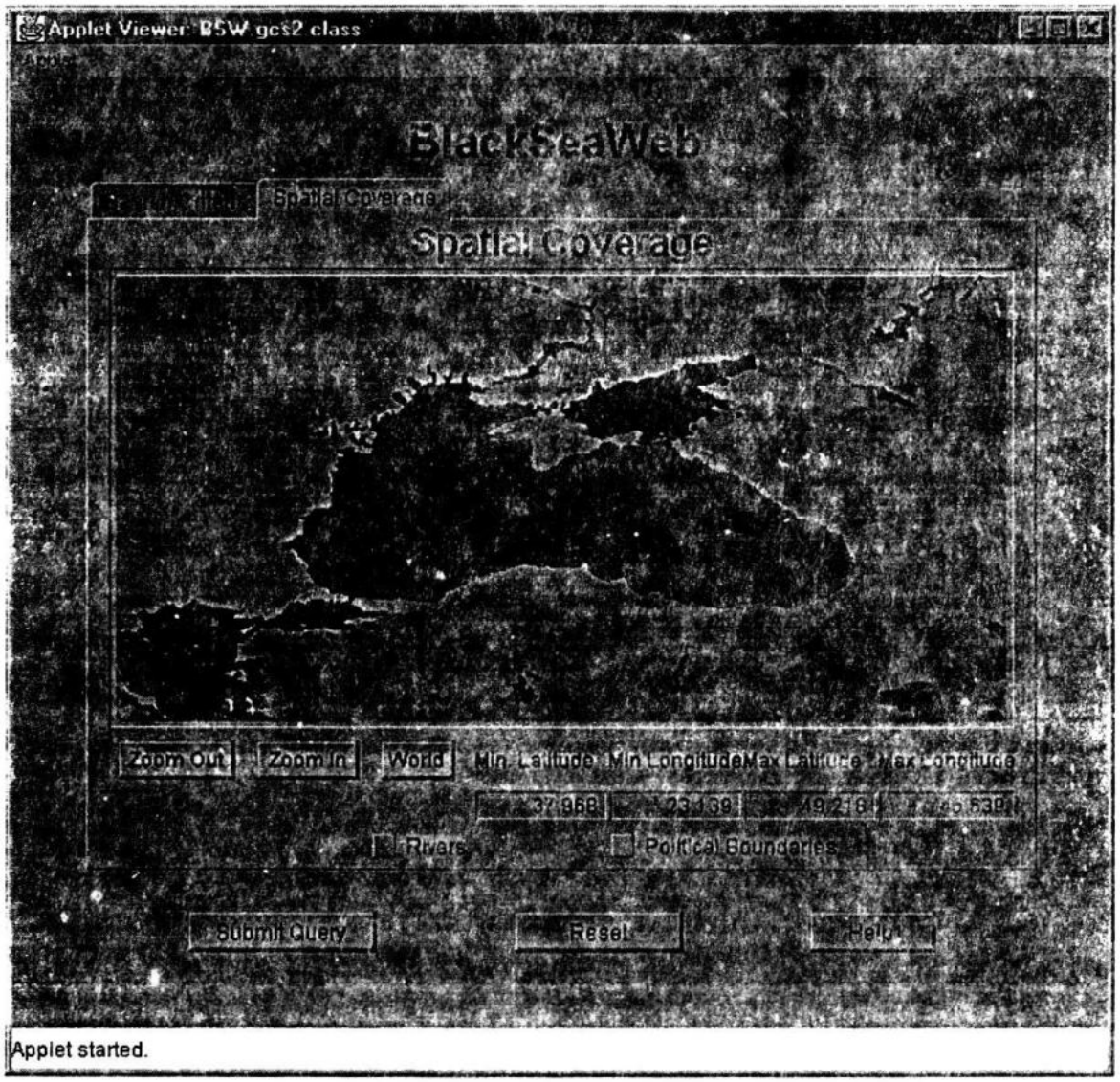

Figure 4. The Spatial Coverage page of User Interface

The Map is displayed using the Map Bean described above.

Page contains the New Query button is used to start formulating a new query. The Fetch Dataset button will start a transfer of the selected stations and profiles to the user platform where the user can save the data and use them in a well know data analysis tool.

CORBA-Server objects

Architectural Model of the **Black Sea Web** (see **Figure 1**) contains 3 types of CORBA Server objects:

- *Central Guidance Catalogue Server object;*
- *Guidance Catalogue Server object;*
- *Local Query Server object.*

The Central Guidance Catalogue Server object is used to maintain the Guidance Catalogue and to distribute changes to the Central Guidance Catalogue to the copies of the Central Guidance Catalogue held at the Local Servers. The Central Guidance Catalogue Server supports the following interface functions:

checkVersion - checks the current version of the GC.

notifyUpdate -the CGCS must request new information from the calling LQS.

In addition to these functions the Central Guidance Catalogue Server object inherits the methods getFields, getValues and getServers from the Guidance Catalogue Server object.

The Guidance Catalogue Server object supports the Java applet when it constructs the user interface. It maintains the Guidance Catalogues on the LQSs, supports the mirroring

of the GC between the CGCS and the GCS located at the different organisations. The Guidance Catalogue Server object supports the following methods:

getFields - get common field names

getValues - get values for a common field

getServers - get information about an LQS

notifyUpdate - server must request new information from the CGCS

The Local Query Server object processes queries generated by the Scientific Client Java Applet. For this purpose the following two methods are provided.

getMetadata - execute the Uniform Query and retrieves the metadata for the found data.

getData - retrieves data for the station(s) or profile(s).

The Uniform Query Engine is the main part of the Local Query Server Object. It implements the getData and getMetadata methods providing conversions of Uniform Query parameters to Local Query and local result to a Uniform Query Result. The Uniform Query Result is then returned to the calling application.

HTTP and CORBA servers

The Central Guidance Catalogue Server is responsible for the maintenance of the Central Guidance Catalogue and the replication of the Central Guidance Catalogue to the Local Servers, which are maintaining their own part of the Guidance Catalogue. The Guidance Catalogue (GC) contains descriptive information (metadata) for each of the Local Query Servers it is supporting, as well as information about the Local Query Servers themselves. The metadata is used by the web page construction module to build the User Interface pages

The purpose of *the Local Server* is to provide the Scientific Client Java Applet with Guidance Catalogue (GC) information, to process queries from the SCJA and to send a result back to the SCJA. The query processing involves on one hand the translation of the Uniform Query (UQ) into a local request and on the other hand the translation of the local answer into a Uniform Query Result.

The Local Server contains:

- Local Guidance Catalogue Server Object (LGCSO);
- Local Guidance Catalogue (LGC);
- Local Query Server Object (LQSO);
- Local Scientific Database (LQDB);
- Mapping Catalogue (MP).

The LGCSO and LQS carry out different tasks and, as a matter of fact, can be located at different hosts. As mentioned above the main reason to locate these servers at one host is an attempt to take into account the bad quality of Internet communication in the Black Sea countries and to improve the **Black Sea Web** distributed database performance.

The local Scientific Database contains environmental metadata and measured data and is a part of the distributed database. The local database is queried through the Uniform Query Engine and the result is returned to the Scientific Client Java Applet.

The purpose of the Local Guidance Catalogue is to enable the Scientific Client Java Applet to build query pages with information, which is available from the Local Query Servers.

The Mapping Catalogue contains description of all Uniform Query Parameters in terms of the Local Database. It is used by the Local Query Server Object to transform Uniform Query to the Local Query. When the local query has been executed and a result generated, the local query result will be processed by the LQSO with using of Mapping Catalogue to convert local fields and values to common fields and values.

Conclusion

Black Sea Marine Environmental Management Support System of the **Black Sea Web** project implements described above architecture, using benefits of ***CORBA/Java*** technologies:

- An open standard.
- Hardware and OS platforms independence. Distributed objects can run on any hardware or software platform.
- Programming language independence.
- Location Transparent.
- All benefits of object-oriented technology.
- Inherent scalability. Distributed object architecture provides inherent scalability since client and server objects can live anywhere.

The main facility of System - Uniform Query interface is implemented in CORBA Scientific Client Java Applet. It is provided to the users through a standard Java capable WWW browser. The Uniform Query interface will enable scientific users to search for data, located at the different data centres without having any knowledge of the individual query languages and/or the organisation of the data centres, which holds the data.

The Uniform Query interface will make it possible for scientists to search for data based on the data type, geographical location and time together with other relevant parameters. Using of

CORBA technology allows to organise data exchange between different physical databases via Internet independently on their type and platform. It is seen as very important that the data centres should not convert their data to a common format but rather continue to use their existing format.

Server side CORBA objects are realised on Java programming language, so they are platform independent. As since Guidance Catalogue Server object is of universal structure, it can be deployed to every new participant of distributed environmental database system as a ready to use tool. The Local Query Server Object, which is responsible for communication with Local Database, also is intended to be universal CORBA object. But in reality it is impossible to implement all variations of database structure in one application. As since the interface of exchange between all CORBA objects is determined, new participants of the distributed environmental database system can choose the most appropriate operating system, execution environment and programming language to use for each component of a system under construction. More importantly, in an ORB-based solution the integration of existing components is allowed. Developers simply needs to model the legacy component using the same IDL they use for creating new objects, then write "wrapper" code that translates between the standardised bus and the legacy interfaces.

Using of ***CORBA/Java*** technologies in **Black Sea Web** project results the Black Sea Marine Environmental Management Support System is a very powerful, flexible and scalable tool for accessing of environmental databases via Internet.