A Browser Front End For CORBA Objects

Atul Kumar, Deepak Gupta, Pankaj Jalote
Department of Computer Science and Engineering, Indian Institute of Technology
Kanpur − 208016, India
{atul, deepak, jalote} @iitk.ac.in

ABSTRACT

We propose a URI scheme for addressing CORBA objects. A URI for an object not only identifies the object but may also optionally include the name of the method to be invoked on the object and the parameters required. We have implemented the URI scheme by extending Sun Microsystems' HotJava browser. With this kind of integration, different web services can be described using CORBA, thereby making the Web extensible for different needs and bringing the richness of distributed objects to the web.

Keywords

URI Scheme for Objects, Object Browser

1. INTRODUCTION

We present an approach for making distributed objects available on the web as a first step towards integrating distributed objects and the web. Our approach is to access the CORBA objects from a browser directly without sending the request to a web server. No CGI wrapping for the CORBA applications is required if a browser can use IIOP to communicate with CORBA ORBs. We have designed an Uniform Resource Identifier (URI) scheme that can address CORBA objects. A browser has been implemented that, given an URI for a CORBA object, locates the object and accesses the Interface Repository associated with the object to get its interface information. CORBA Dynamic Invocation Interface (DII) is then used to invoke a method on the object if the method name and parameters are specified in the URI.

2. URI SCHEME FOR CORBA OBJECTS

Two new naming schemes starting with ior:// and iiopname:// are added to the existing set of naming schemes for URIs. ior:// is used when an object is specified by its stringified Interoperable Object Reference (IOR). iiopname:// is taken from the Interoperable Naming Service proposal submitted to the OMG by four organizations [1].

The hexadecimal strings for `ior' URIs are generated by first turning an object reference into an IOR, and then encapsulating the IOR using the encoding rules specified in GIOP version 1.0. The content of the encapsulated IOR is then turned into hexadecimal digit pairs. An example of the `ior' scheme is: ior://0032A3CF....

An `iiopname' URI contains an address and an object name. Address consists of some DNS host name (or IP address) and a TCP port number. Object name consists of a hierarchical CORBA name. An example of `iiopname' scheme is: iiopname://somehost.com/a/string/path/to/obj. This URI specifies that at host somehost.com, an object of type NamingContext (with an object key of NameService) can be found. The string a/string/path/to/obj can then be used to obtain the object reference.

Method name follows the object location part of the URI. Two consecutive colons (::) are used as separator between the object location part and the method name. Following is an example URI with method name: iiopname://objectserver.com/object/name::do_job

We use a scheme similar to the CGI for passing the parameter values to the method. The URI with parameter values looks like: iiopname://objectserver.com/one/two/object::do_job?par1=val1
Above scheme does not require the parameters to be passed in the same order as defined in the IDL interface. This is because the parameter name is included in the URI with its value. However, some ambiguity may exist if the method is overloaded. Adding parameter type will remove such ambiguity.

3. OBJECT BROWSER

A prototype browser for accessing CORBA objects is developed by extending HotJava web browser [2]. New protocols (naming schemes) can be added to HotJava by writing Protocol Handlers [3]. Our implementation assumes that an interface repository is available to the object and its IDL interface is registered with the interface repository.

Figure 1 shows the page generated by the browser for an example URI. This page contains HTML forms for all methods defined in the example interface named `Bank'.

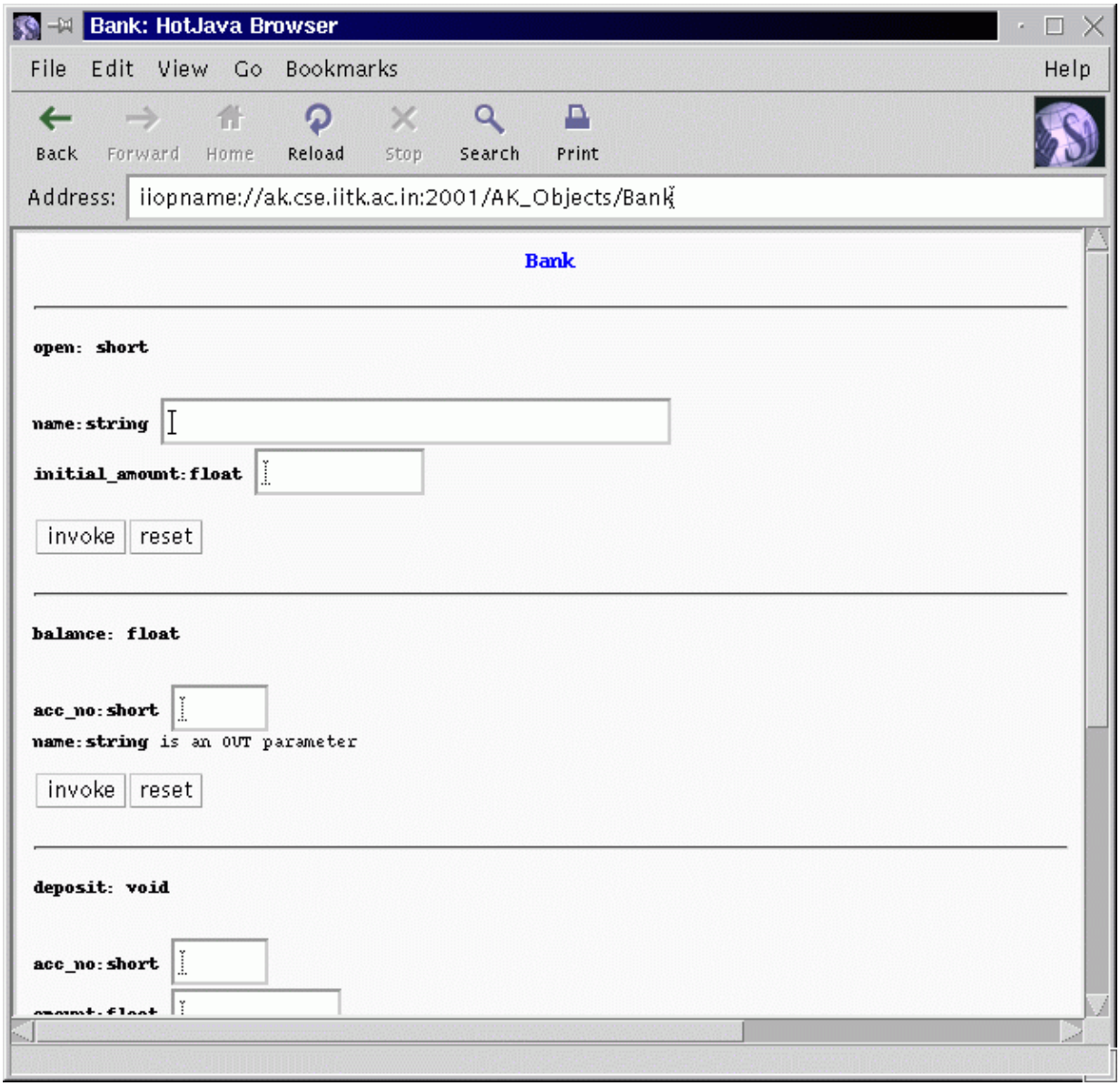

Figure 1 − Page containing forms for all methods of interface `Bank'

4. REFERENCES

1. Interoperable Naming Service, Joint Revised Submission by BEA Systems, DSTC, IONA, and Inprise, OMG Document orbos/98−10−11.
2. HotJava Browser 3.0. URL: http://java.sun.com/products/hotjava/3.0/
3. Mark Wutka, et. al., Adding Additional Protocols to HotJava, JAVA Expert Solutions, Chapter 35. URL: http://docs.rinet.ru/JSol/ch35.htm