

Network Working Group
Request for Comments: 2714
Category: Informational

V. Ryan
R. Lee
S. Seligman
Sun Microsystems, Inc.
October 1999

Schema for Representing CORBA Object References in an LDAP Directory

Status of this Memo

This memo provides information for the Internet community. It does not specify an Internet standard of any kind. Distribution of this memo is unlimited.

Copyright Notice

Copyright (C) The Internet Society (1999). All Rights Reserved.

Abstract

CORBA [CORBA] is the Common Object Request Broker Architecture defined by the Object Management Group. This document defines the schema for representing CORBA object references in an LDAP directory [LDAPv3].

1. Introduction

This document assumes that the reader has a general understanding of CORBA.

Traditionally, LDAP directories have been used to store data. Users and programmers think of the directory as a hierarchy of directory entries, each containing a set of attributes. You look up an entry from the directory and extract the attribute(s) of interest. For example, you can look up a person's telephone number from the directory. Alternatively, you can search the directory for entries with a particular set of attributes. For example, you can search for all persons in the directory with the surname "Smith".

CORBA applications require access to CORBA objects. Traditionally, CORBA applications have used the COS Naming service for storage and retrieval of CORBA object references. When deployed in environments with a directory, CORBA applications should be able to use the directory as a repository for CORBA object references. The directory provides a centrally administered, and possibly replicated, service for use by CORBA applications distributed across the network.

For example, an application server may use the directory for "registering" CORBA objects representing the services that it manages, so that a client can later search the directory to locate those services as it needs.

The motivation for this document is to define a common way for applications to store and retrieve CORBA object references from the directory. Using this common schema, any CORBA application that needs to read or store CORBA object references in the directory can do so in an interoperable way.

Note that this schema is defined for storing CORBA "object references," not CORBA objects in general. There might be other ways to store CORBA objects in an LDAP directory but they are not covered by this schema.

2. Representation of CORBA Object References

This document defines schema elements to represent a CORBA object reference in LDAP directory. Applications in possession of a reference to an object can invoke calls on that object. Such a reference is termed an "interoperable object reference," or IOR. Access to CORBA objects by using IORs is achieved transparently to the application, by means of the General Inter-ORB Protocol.

A CORBA object reference is represented in the directory by the object class `corbaObjectReference`. `corbaObjectReference` is a subclass of the abstract `corbaObject` object class. `corbaObjectReference` is an auxiliary object class, which means that it needs to be mixed in with a structural object class.

The object class `corbaContainer` is used in a directory entry which represents a CORBA object or object reference. It is a structural object class, and when representing an object reference, the `corbaObjectReference` object class would also need to be present in the entry. `corbaContainer` is not required when a subclass of `corbaObject` (such as `corbaObjectReference`) is mixed in with another structural object class.

The definitions for the object classes `corbaObject`, `corbaObjectReference`, and `corbaContainer` are presented in Section 4.

The `corbaObject` class has two optional attributes: `corbaRepositoryId` and `description`. `corbaRepositoryId` is a multivalued attribute that is used to store the repository ids of the interfaces implemented by a CORBA object. `description` is used to store a textual description of a CORBA object.

The `corbaObjectReference` class has one mandatory attribute: `corbaIor`. `corbaIor` is used to store the object's stringified IOR.

`corbaIor` and `corbaRepositoryId` are defined in Section 3; description is defined in [v3Schema].

3. Attribute Type Definitions

The following attribute types are defined in this document:

```
corbaIor
corbaRepositoryId
```

3.1 `corbaIor`

This attribute stores the string representation of the interoperable object reference (IOR) for a CORBA object. An IOR is an opaque handle for the object which contains the information necessary to locate the object, even if the object is in another ORB.

This attribute's syntax is 'IA5 String' and its case is insignificant.

```
( 1.3.6.1.4.1.42.2.27.4.1.14
  NAME 'corbaIor'
  DESC 'Stringified interoperable object reference of a CORBA object'
  EQUALITY caseIgnoreIA5Match
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.26
  SINGLE-VALUE
)
```

3.2 `corbaRepositoryId`

Each CORBA interface has a unique "repository id" (also called "type id") that identifies the interface. A CORBA object has one or more repository ids, one for each interface that it implements.

The format of a repository id can be any string, but the OMG specifies four standard formats:

a. IDL-style

```
IDL:Prefix/ModuleName/InterfaceName:VersionNumber
```

For example, the repository id for the "NamingContext" in OMG's COS Naming module is: "IDL:omg.org/CosNaming/NamingContext:1.0".

b. RMI-style

```
RMI:ClassName:HashCode[:SUID]
```

This format is used by RMI-IIOP remote objects [RMI-IIOP]. "ClassName" is the fully qualified name of the class (for example, "java.lang.String"). "HashCode" is the object's hash code (that is, that obtained by invoking the "hashCode()" method). "SUID" is the "stream unique identifier", which is a 64-bit number that uniquely identifies the serialization version of the class; SUID is optional in the repository id.

c. DCE-style

```
DCE:UUID
```

This format is used for DCE/CORBA interoperability [CORBA-DCE]. "UUID" represents a DCE UUID.

d. "local"

This format is defined by the local Object Request Broker (ORB).

The corbaRepositoryId attribute is a multivalued attribute; each value records a single repository id of an interface implemented by the CORBA object. This attribute need not contain a complete list of the interfaces implemented by the CORBA object.

This attribute's syntax is 'Directory String' and its case is significant. The values of this attribute are encoded using UTF-8. Some values may require translation from their native representation in order to be correctly encoded using UTF-8.

```
( 1.3.6.1.4.1.42.2.27.4.1.15
  NAME 'corbaRepositoryId'
  DESC 'Repository ids of interfaces implemented by a CORBA object'
  EQUALITY caseExactMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
)
```

4. Object Class Definitions

The following object classes are defined in this document:

```
corbaContainer
corbaObject
corbaObjectReference
```

4.1 corbaContainer

This structural object class represents a container for a CORBA object.

```
( 1.3.6.1.4.1.42.2.27.4.2.10
  NAME 'corbaContainer'
  DESC 'Container for a CORBA object'
  SUP top
  STRUCTURAL
  MUST ( cn )
)
```

4.2 corbaObject

This abstract object class is the root class for representing a CORBA object.

```
( 1.3.6.1.4.1.42.2.27.4.2.9
  NAME 'corbaObject'
  DESC 'CORBA object representation'
  SUP top
  ABSTRACT
  MAY ( corbaRepositoryId $ description )
)
```

4.3 corbaObjectReference

This auxiliary object class represents a CORBA object reference. It must be mixed in with a structural object class.

```
( 1.3.6.1.4.1.42.2.27.4.2.11
  NAME 'corbaObjectReference'
  DESC 'CORBA interoperable object reference'
  SUP corbaObject
  AUXILIARY
  MUST ( corbaIor )
)
```

5. Security Considerations

Obtaining a reference to an object and storing it in the directory may make a handle to the object available to a wider audience. This may have security implications.

6. Acknowledgements

We would like to thank Sanjeev Krishnan of Sun Microsystems, Simon Nash of IBM, and Jeffrey Spirn of Oracle for their comments and suggestions.

7. References

- [CORBA] The Object Management Group, "Common Object Request Broker Architecture Specification 2.2", <http://www.omg.org>
- [CORBA-DCE] Distributed Systems Technology Center and Digital Equipment Corporation, "DCE/CORBA Interworking Specification", May 1998. http://www.omg.org/library/schedule/DCE_CORBA_Interworking_RFP.html
- [LDAPv3] Wahl, M., Howes, T. and S. Kille, "Lightweight Directory Access Protocol (v3)", RFC 2251, December 1997.
- [RMI-IIOP] IBM and Java Software, Sun Microsystems, Inc., "RMI over IIOP", June 1999. <http://java.sun.com/products/rmi-iiop/index.html>
- [v3Schema] Wahl, M., "A Summary of the X.500(96) User Schema for use with LDAPv3", RFC 2256, December 1997.

8. Authors' Addresses

Vincent Ryan
Sun Microsystems, Inc.
Mail Stop EDUB03
901 San Antonio Road
Palo Alto, CA 94303
USA

Phone: +353 1 819 9151
EMail: vincent.ryan@ireland.sun.com

Rosanna Lee
Sun Microsystems, Inc.
Mail Stop UCUP02-206
901 San Antonio Road
Palo Alto, CA 94303
USA

Phone: +1 408 863 3221
EMail: rosanna.lee@eng.sun.com

Scott Seligman
Sun Microsystems, Inc.
Mail Stop UCUP02-209
901 San Antonio Road
Palo Alto, CA 94303
USA

Phone: +1 408 863 3222
EMail: scott.seligman@eng.sun.com

9. Appendix - LDAP Schema

```
-- Attribute types --
```

```
( 1.3.6.1.4.1.42.2.27.4.1.14
  NAME 'corbaIor'
  DESC 'Stringified interoperable object reference of a CORBA object'
  EQUALITY caseIgnoreIA5Match
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.26
  SINGLE-VALUE
)
```

```
( 1.3.6.1.4.1.42.2.27.4.1.15
  NAME 'corbaRepositoryId'
  DESC 'Repository ids of interfaces implemented by a CORBA object'
  EQUALITY caseExactMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
)
```

```
-- from RFC-2256 --
```

```
( 2.5.4.13
  NAME 'description'
  EQUALITY caseIgnoreMatch
  SUBSTR caseIgnoreSubstringsMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{1024}
)
```

```
-- Object classes --
```

```
( 1.3.6.1.4.1.42.2.27.4.2.9
  NAME 'corbaObject'
  DESC 'CORBA object representation'
  SUP top
  ABSTRACT
  MAY ( corbaRepositoryId $ description )
)
```

```
( 1.3.6.1.4.1.42.2.27.4.2.10
  NAME 'corbaContainer'
  DESC 'Container for a CORBA object'
  SUP top
  STRUCTURAL
  MUST ( cn )
)
```

```
( 1.3.6.1.4.1.42.2.27.4.2.11
  NAME 'corbaObjectReference'
  DESC 'CORBA interoperable object reference'
  SUP corbaObject
  AUXILIARY
  MUST ( corbaIor )
)
```

-- Matching rule from ISO X.520 --

```
( 2.5.13.5
  NAME 'caseExactMatch'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
)
```

10. Full Copyright Statement

Copyright (C) The Internet Society (1999). All Rights Reserved.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to the Internet Society or other Internet organizations, except as needed for the purpose of developing Internet standards in which case the procedures for copyrights defined in the Internet Standards process must be followed, or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by the Internet Society or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.

