

Fault Tolerant CORBA Extensions for JINI Pattern Language

Aniruddha Gokhale
Bell Laboratories
Lucent Technologies
600 Mountain Avenue
Murray Hill, NJ 07974
agokhale@lucent.com

Abstract

An increasing number of applications are being developed using the Jini pattern language. The Jini pattern language comprises patterns present in Sun Microsystems' Jini technology as well as other *ad hoc* networking technologies. The Jini pattern language, however, is still evolving. The primary areas of evolution include identifying additional patterns for *ad hoc* networking, load balancing and fault tolerance.

This paper proposes simple extensions to the Object Management Group's Fault Tolerant CORBA standard (FT-CORBA) to make it applicable to the Jini pattern language.

1 Motivation

There is a trend, due primarily to recent advances in *ad hoc* networking, towards developing applications based on the Jini pattern [1] language. The Jini pattern language is an evolving paradigm comprising the patterns used by *ad hoc* networking technologies such as Sun Microsystems' Jini [2] architecture, Bluetooth [3] and others, and the patterns used for load balancing, performance optimizations, and fault-tolerance.

We propose simple extensions to the Object Management Group's Fault Tolerant CORBA standard (FT-CORBA) [4] for it to be incorporated in the Jini pattern language.

This paper is organized as follows: Section 2 provides a brief overview of the FT-CORBA standard; Section 3 describes our proposed extensions to the FT-CORBA standard; and finally Section 4 provides concluding remarks.

2 Overview of the Fault Tolerant CORBA Specification

The Fault Tolerant CORBA (FT-CORBA) [4] specification defines a standard set of interfaces, policies, and services that

provide robust support for applications requiring high reliability. The fault tolerance mechanism used to detect and recover from failures is based on *entity redundancy*. Naturally, in FT-CORBA the redundant entities are replicated CORBA objects.

Figure 1 illustrates the key components in the FT-CORBA

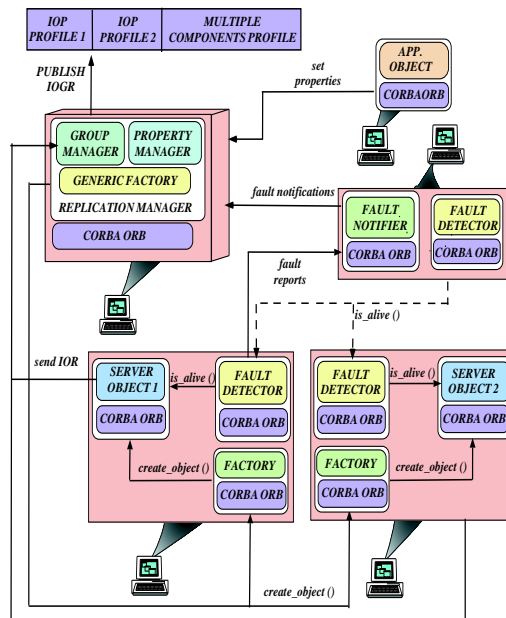


Figure 1: The Architecture of Fault Tolerant CORBA

architecture. All components shown in the figure are implemented as standard CORBA objects, *i.e.*, they are defined using CORBA IDL interfaces and implemented using servants that can be written in standard programming languages, such as Java, C++, C, or Ada.

The functionality of relevant components is described below.

Interoperable object group references (IOGRs): FT-CORBA standardizes the format of interoperable object group references (IOGR) used for a replica group. FT-CORBA

servers register their IOGRs with a lookup [5] service. Clients use these IOGRs to invoke operations on servers.

ReplicationManager: This component is responsible for managing replicas which includes creation and recovery.

Fault Detector and Notifier: `FaultDetectors` are CORBA objects responsible for detecting faults via either a *pull-based* or a *push-based* mechanism. `FaultDetectors` report faults to `FaultNotifiers`. In turn, the `FaultNotifiers` propagate these notifications to a `ReplicationManager`, which performs recovery actions.

FT-CORBA is designed to prevent single points of failure within a distributed object computing system. As a result, each component described above must itself be replicated. Moreover, mechanisms must be provided to deal with potential failures and recovery.

3 Enabling FT-CORBA for the Jini Paradigm

Services operating in *ad hoc* networking environments such as Jini register themselves with a lookup [5] service and request a lease [6] indicating the time they must be registered with the lookup service.

The FT-CORBA standard in its current form has no notion of a lease when a replica group is created and monitored for faults. This shortcoming makes the FT-CORBA standard unsuitable for *ad hoc* networks where services are mobile. Thus, for the patterns in FT-CORBA to be incorporated into the Jini pattern language requires extensions to the FT-CORBA standard. The extensions we propose preserve FT-CORBA's client-side transparent failovers in the event of server failures.

In the following sections, we describe two extensions to FT-CORBA.

Adding Lease to FT-CORBA Object Group creation We propose extending FT-CORBA to include the notion of lease for the monitored replica objects. To achieve this, we require that the `GenericFactory`'s `create_object` operation, which creates a replica group, additionally request a lease indicating the time the replica group should be monitored for faults. The `ReplicationManager` will then instruct the `FaultDetectors` to monitor the replicas only until the lease is valid. Any failures detected before the lease has expired will involve recovery and clients will experience transparent failovers as before. Otherwise the replica group is considered to have moved and no recovery is attempted. In such a situation, the client-side ORB will receive a `LeaseExpired` exception.

Extending the IOGR and the ReplicationManager The IOGR standardized by FT-CORBA comprises the interoperable object references (IORs) of individual replicas belonging to the replica group. We propose extending the IOGR to include the IOR of the `ReplicationManager`. This additional IOR is used as a fallback object for the clients to reach in the event that a replica group's lease has expired and the group has moved away.

When a client makes a request to a server group whose lease has expired, the request will end up at the `ReplicationManager` since none of the replica IORs in the IOGR would be valid. In such an event, the `ReplicationManager` can be programmed to perform a lookup for the service and possibly a new `ReplicationManager` who manages the group. If it finds a new service based on the properties requested earlier by the client, it can create a new IOGR and throw a `LOCATION_FORWARD` exception to the client-side ORB. The client can then start communication with the newly found service. If such a service cannot be located, then the client will receive a `LeaseExpired` exception.

The underlying assumption in the above enhancement is that the `ReplicationManagers` are not mobile *i.e.*, they will typically be hosted on *base stations*.

A variant of this idea would be to have the client-side ORB do a new lookup for the service if it receives a `LeaseExpired` exception. However, this scheme increases the client-side ORB complexity which is not desirable.

4 Concluding Remarks

This paper proposes two extensions to FT-CORBA to enable it to be used in *ad hoc* networks. Our extensions preserve the FT-CORBA client-side transparent failover features. In addition, they address the mobility issues of the services while providing them fault tolerance.

References

- [1] Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides, *Design Patterns: Elements of Reusable Object-Oriented Software*, Addison-Wesley, Reading, MA, 1995.
- [2] Sun Microsystems, "Jini Connection Technology," <http://www.sun.com/jini/index.html>, 1999.
- [3] Bluetooth Consortium, "Bluetooth Wireless Technology Specification v1.0 Core," http://www.bluetooth.com/developer/specification/core_10_b.pdf, 1999.
- [4] Object Management Group, *Fault Tolerant CORBA Specification*, OMG Document orbos/99-12-08 edition, December 1999.
- [5] Michael Kircher and Prashant Jain, "Lookup Pattern," in *Submitted to EuroPloP 2000 conference*, Irsee, Germany, July 2000.
- [6] Prashant Jain and Michael Kircher, "Leasing Pattern," in *PloP 2000 conference*, Allerton Park, IL, August 2000.