Date: August 2007

Lightweight Load Balancing Service, Beta 1

OMG Adopted Specification

OMG Document Number: ptc/2007-08-11

This OMG document replaces the submission document (mars/2007-03-11, Alpha). It is an OMG Adopted Beta Specification and is currently in the finalization phase. Comments on the content of this document are welcome, and should be directed to *issues@omg.org* by December 21, 2007.

You may view the pending issues for this specification from the OMG revision issues web page *http://www.omg.org/issues/*.

The FTF Recommendation and Report for this specification will be published on July 4, 2008. If you are reading this after that date, please download the available specification from the OMG Specifications Catalog.

USE OF SPECIFICATION - TERMS, CONDITIONS & NOTICES

LICENSES

PATENTS

GENERAL USE RESTRICTIONS

DISCLAIMER OF WARRANTY

WHILE THIS PUBLICATION IS BELIEVED TO BE ACCURATE, IT IS PROVIDED "AS IS" AND MAY CONTAIN ERRORS OR MISPRINTS. THE OBJECT MANAGEMENT GROUP AND THE COMPANIES LISTED ABOVE MAKE NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS PUBLICATION, INCLUDING BUT NOT LIMITED TO ANY WARRANTY OF TITLE OR OWNERSHIP, IMPLIED WARRANTY OF MERCHANTABILITY OR WARRANTY OF FITNESS FOR A PARTICULAR PURPOSE OR USE. IN NO EVENT SHALL THE OBJECT MANAGEMENT GROUP OR ANY OF THE COMPANIES LISTED ABOVE BE LIABLE FOR ERRORS CONTAINED HEREIN OR FOR DIRECT, INDIRECT, INCIDENTAL, SPECIAL, CONSEQUENTIAL, RELIANCE OR COVER DAMAGES, INCLUDING LOSS OF PROFITS, REVENUE, DATA OR USE, INCURRED BY ANY USER OR ANY THIRD PARTY IN CONNECTION WITH THE FURNISHING, PERFORMANCE, OR USE OF THIS MATERIAL, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

The entire risk as to the quality and performance of software developed using this specification is borne by you. This disclaimer of warranty constitutes an essential part of the license granted to you to use this specification.

RESTRICTED RIGHTS LEGEND

Use, duplication or disclosure by the U.S. Government is subject to the restrictions set forth in subparagraph (c) (1) (ii) of The Rights in Technical Data and Computer Software Clause at DFARS 252.227-7013 or in subparagraph (c)(1) and (2) of the Commercial Computer Software - Restricted Rights clauses at 48 C.F.R. 52.227-19 or as specified in 48 C.F.R. 227-7202-2 of the DoD F.A.R. Supplement and its successors, or as specified in 48 C.F.R. 12.212 of the Federal Acquisition Regulations and its successors, as applicable. The specification copyright owners are as indicated above and may be contacted through the Object Management Group, 140 Kendrick Street, Needham, MA 02494, U.S.A.

TRADEMARKS

MDA®, Model Driven Architecture®, UML®, UML Cube logo®, OMG Logo®, CORBA® and XMI® are registered trademarks of the Object Management Group, Inc., and Object Management Group™, OMG™ , Unified Modeling Language™, Model Driven Architecture Logo™, Model Driven Architecture Diagram™, CORBA logos™, XMI Logo™, CWM™, CWM Logo™, IIOP™ , MOF™ , OMG Interface Definition Language (IDL)™, and OMG SysML™ are trademarks of the Object Management Group. All other products or company names mentioned are used for identification purposes only, and may be trademarks of their respective owners.

COMPLIANCE

The copyright holders listed above acknowledge that the Object Management Group (acting itself or through its designees) is and shall at all times be the sole entity that may authorize developers, suppliers and sellers of computer software to use certification marks, trademarks or other special designations to indicate compliance with these materials.

Software developed under the terms of this license may claim compliance or conformance with this specification if and only if the software compliance is of a nature fully matching the applicable compliance points as stated in the specification. Software developed only partially matching the applicable compliance points may claim only that the software was based on this specification, but may not claim compliance or conformance with this specification. In the event that testing suites are implemented or approved by Object Management Group, Inc., software developed using this specification may claim compliance or conformance with the specification only if the software satisfactorily completes the testing suites.

OMG’s Issue Reporting Procedure

All OMG specifications are subject to continuous review and improvement. As part of this process we encourage readers to report any ambiguities, inconsistencies, or inaccuracies they may find by completing the Issue Reporting Form listed on the main web page *http://www.omg.org*, under Documents, Report a Bug/Issue (http://www.omg.org/technology/agreement.htm).

Table of Contents

Preface

About the Object Management Group

OMG

Founded in 1989, the Object Management Group, Inc. (OMG) is an open membership, not-for-profit computer industry standards consortium that produces and maintains computer industry specifications for interoperable, portable and reusable enterprise applications in distributed, heterogeneous environments. Membership includes Information Technology vendors, end users, government agencies and academia.

OMG member companies write, adopt, and maintain its specifications following a mature, open process. OMG's specifications implement the Model Driven Architecture® (MDA®), maximizing ROI through a full-lifecycle approach to enterprise integration that covers multiple operating systems, programming languages, middleware and networking infrastructures, and software development environments. OMG's specifications include: UML® (Unified Modeling Language™); CORBA® (Common Object Request Broker Architecture); CWM™ (Common Warehouse Metamodel); and industry-specific standards for dozens of vertical markets.

More information on the OMG is available at *http://www.omg.org/*.

OMG Specifications

As noted, OMG specifications address middleware, modeling and vertical domain frameworks. A catalog of all OMG Specifications is available from the OMG website at:

http://www.omg.org/technology/documents/spec_catalog.htm

Specifications within the Catalog are organized by the following categories:

OMG Modeling Specifications

- UML
- MOF
- XMI
- CWM
- Profile specifications

OMG Middleware Specifications

- CORBA/IIOP
- IDL/Language Mappings
- Specialized CORBA specifications
- CORBA Component Model (CCM)

Platform Specific Model and Interface Specifications

- CORBAservices

- CORBAfacilities
- OMG Domain specifications
- OMG Embedded Intelligence specifications
- OMG Security specifications

All of OMG's formal specifications may be downloaded without charge from our website. (Products implementing OMG specifications are available from individual suppliers.) Copies of specifications, available in PostScript and PDF format, may be obtained from the Specifications Catalog cited above or by contacting the Object Management Group, Inc. (as of January 16, 2006) at:

OMG Headquarters
140 Kendrick Street
Building A, Suite 300
Needham, MA 02494
USA
Tel: +1-781-444-0404
Fax: +1-781-444-0320
Email: *pubs@omg.org*

Certain OMG specifications are also available as ISO standards. Please consult *http://www.iso.org*

Typographical Conventions

The type styles shown below are used in this document to distinguish programming statements from ordinary English. However, these conventions are not used in tables or section headings where no distinction is necessary.

Times/Times New Roman - 10 pt.: Standard body text

Helvetica/Arial - 10 pt. Bold: OMG Interface Definition Language (OMG IDL) and syntax elements.

Courier - 10 pt. Bold: Programming language elements.

Helvetica/Arial - 10 pt: Exceptions

Note – Terms that appear in *italics* are defined in the glossary. Italic text also represents the name of a document, specification, or other publication.

Issues

The reader is encouraged to report any technical or editing issues/problems with this specification to *http://www.omg.org/technology/agreement.htm*.

1 Scope

The concepts at the foundation of the Lightweight Load Balancing Service are in a sense minimal and general, thus providing a good abstraction for many different platforms. The key features provided by the service can be divided in the following categories.

Group Management. The management of server replicas leverages from existing OMG standards, and thus relies on the Portable Group.

Load Balancing Service. The Lightweight Load Balancing Service provides a framework for, per-request, static load balancing. The standard allows the definition of user defined load balancing strategies, while requiring complying implementations to provide at least the round robin and random strategies. This service provides a façade for managing load balancing strategies, along with the creation of load balancing groups. It is worth mentioning that some implementations might choose to implement the Load Balancing Service and the Group Management by means of a single entity, as outlined in Figure 1, however, this specification intentionally decouples the role of group manager from that of the load balancing service, as they really have different roles.

Load Balancing Strategies. Load balancing strategies implement the logic for selecting which server replica has to serve the next client request. This standard provides APIs for static load balancing only, however there is nothing in the current API which inhibits future extensions that might include adaptive load balancing.

2 Compliance

2.1 Compliance Levels

This is a CORBA Service specification that forms a single optional conformance point that operates within the context of a conforming CORBA implementation.

3 Normative References

The following normative documents contain provisions which, through reference in this text, constitute provisions of this specification. For dated references, subsequent amendments to, or revisions of, any of these publications do not apply.

- [CORBA] Common Object Request Broker Architecture: Core Specification, OMG, V3.0.3 (formal/04-03-12).
- [DPC] Data Parallel CORBA Specification, OMG, (formal/06-01-03)

4 Terms and Definitions

4.1 General Definitions

- Architecture Board (AB) - The OMG plenary that is responsible for ensuring the technical merit and MDA-compliance of RFPs and their submissions.

- Board of Directors (BoD) - The OMG body that is responsible for adopting technology.
- Common Object Request Broker Architecture (CORBA) - An OMG distributed computing platform specification that is independent of implementation languages.
- Common Warehouse Metamodel (CWM) - An OMG specification for data repository integration.
- CORBA Component Model (CCM) - An OMG specification for an implementation language independent distributed component model.
- Interface Definition Language (IDL) - An OMG and ISO standard language for specifying interfaces and associated data structures.
- Letter of Intent (LOI) - A letter submitted to the OMG BoD's Business Committee signed by an officer of an organization signifying its intent to respond to the RFP and confirming the organization's willingness to comply with OMG's terms and conditions, and commercial availability requirements.
- Mapping - Specification of a mechanism for transforming the elements of a model conforming to a particular metamodel into elements of another model that conforms to another (possibly the same) metamodel.
- Metadata - Data that represents models. For example, a UML model; a CORBA object model expressed in IDL; and a relational database schema expressed using CWM.
- Metamodel - A model of models.
- Meta Object Facility (MOF) - An OMG standard, closely related to UML, that enables metadata management and language definition.
- Model - A formal specification of the function, structure and/or behavior of an application or system.
- Model Driven Architecture (MDA) - An approach to IT system specification that separates the specification of functionality from the specification of the implementation of that functionality on a specific technology platform.
- Normative - Provisions that one must conform to in order to claim compliance with the standard. (as opposed to non-normative or informative which is explanatory material that is included in order to assist in understanding the standard and does not contain any provisions that must be conformed to in order to claim compliance).
- Normative Reference - References that contain provisions that one must conform to in order to claim compliance with the standard that contains said normative reference.
- Platform - A set of subsystems/technologies that provide a coherent set of functionality through interfaces and specified usage patterns that any subsystem that depends on the platform can use without concern for the details of how the functionality provided by the platform is implemented.
- Platform Independent Model (PIM) - A model of a subsystem that contains no information specific to the platform, or the technology that is used to realize it.
- Platform Specific Model (PSM) - A model of a subsystem that includes information about the specific technology that is used in the realization of it on a specific platform, and hence possibly contains elements that are specific to the platform.
- Request for Information (RFI) - A general request to industry, academia, and any other interested parties to submit information about a particular technology area to one of the OMG's Technology Committee subgroups.
- Request for Proposal (RFP) - A document requesting OMG members to submit proposals to the OMG's Technology Committee. Such proposals must be received by a certain deadline and are evaluated by the issuing task force.

- Task Force (TF) - The OMG Technology Committee subgroup responsible for issuing a RFP and evaluating submission(s).
- Technology Committee (TC) - The body responsible for recommending technologies for adoption to the BoD. There are two TCs in OMG - Platform TC (PTC), that focuses on IT and modeling infrastructure related standards; and Domain TC (DTC), that focus on domain specific standards.
- Unified Modeling Language (UML) - An OMG standard language for specifying the structure and behavior of systems. The standard defines an abstract syntax and a graphical concrete syntax.
- UML Profile - A standardized set of extensions and constraints that tailors UML to particular use.
- XML Metadata Interchange (XMI) - An OMG standard that facilitates interchange of models via XML documents.

4.2 Definitions specific to this document

- Load Balancing Service - A CORBA service, defined by this specification, which takes care of distributing incoming requests across a set of server replicas. The distribution logic is specified by means of a Load Balancing Strategy, which might be user defined.
- Load Balancing Strategy - Represents an algorithm used to choose the server replica which will execute the current request. Possible strategies include, round robin, random, etc.

5 Acronyms and Abbreviations

CORBA	Common Object Request Broker Architecture
HTTP	HyperText Transfer Protocol
LB	Load Balancing
MOF	Managed Object Format (not to be confused with Meta Object Facility) - Textual notation used by DMTF to represent CIM models
OMG	Object Management Group
RFP	Request For Proposal
UML	Unified Modeling Language
XML	eXtensible Mark-up Language

6 Acknowledgements

The specification is submitted by:

- SELEX SI

7 Overview

The need for distributing incoming requests across a set of servers, so to share incoming load, appears in many application domains, such as web servers, enterprise information systems, as well as mission and safety critical applications such as Air Traffic Control Systems. While the high level goal-distributing load-is common across all the application domains mentioned above, there are few differences in terms of non-functional requirements, support for dynamic vs. static features which has made very difficult, so far, to agree on a standard Load Balancing Service.

This document has the goal to fill this specification gap, by (1) standardizing a minimal set of interfaces for per request, static load balancing, relevant to as many application domains as possible, and (2) allowing different implementation strategy, so to make it possible to accommodate the different non functional requirements which characterize the application domains aforementioned.

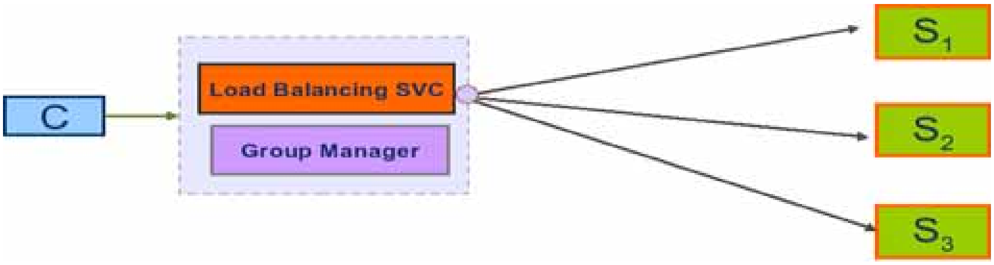

Figure 7.1 - Centralized Load Balancing Service

As an example, some application domains are not so much concerned with availability and performance, and might decide to implement this service as a stand-alone server (see Figure 1), incurring thus in the overhead of a remote CORBA call for each load balancing request. Other, will have to cope with stringent availability and performance requirements and might decide to implement the service as a federation of co-located load balancing service (see Figure 2). The current document leaves room to different implementation architectures as well as techniques (e.g. interceptors vs. locators).

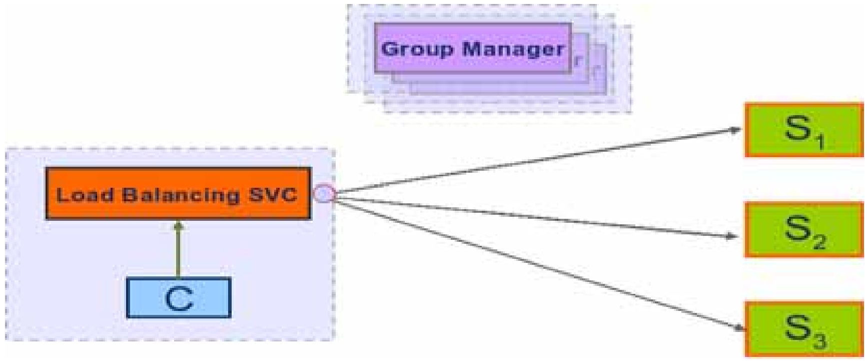

Figure 7.2 - Distributed Load Balancing Service

Finally, it is worth mentioning that the current document represents an effort carried by the CORBA users and vendors communities to define a minimal set of APIs which would guarantee application portability, while allowing vendors to exploit different implementation strategies. Moreover, it has been consciously decided not to impose interoperability across implementations as this was perceived as a potential limiting factor with respect to the specification adoption.

8 Platform Independent Model (PIM)

This part of the document provides a description of the Lightweight Load Balancing Service PIM.

8.1 Packages Organization

The Lightweight Load Balancing Service is organized as a single package with a dependency to the Portable Group for managing load balanced replicas.

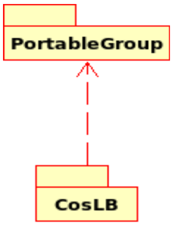

Figure 8.1 - Package

8.2 Group Management

Figure 8.2 shows the UML diagram for the Lightweight Load Balancing Service group manager. As shown in the diagram, the LBGroupManager does not provide new methods for the creation of load balanced replicas, instead it relies on the existing mechanism provided by Criteria for specifying the scheduling policy to be associated with the group. Specifically, the following convention shall be used in order to specify the load balancing strategy:

- Property::nam[0]::id shall always be set to “LBStrategy”
- Property::nam[0]::kind is ignored and should be set to ""
- Property::val shall be a valid strategy name. This specification defines “ROUND_ROBIN” and “RANDOM.” Other strategies might be defined by the user.

When no criteria is provided the default load balancing strategy is the ROUND_ROBIN.

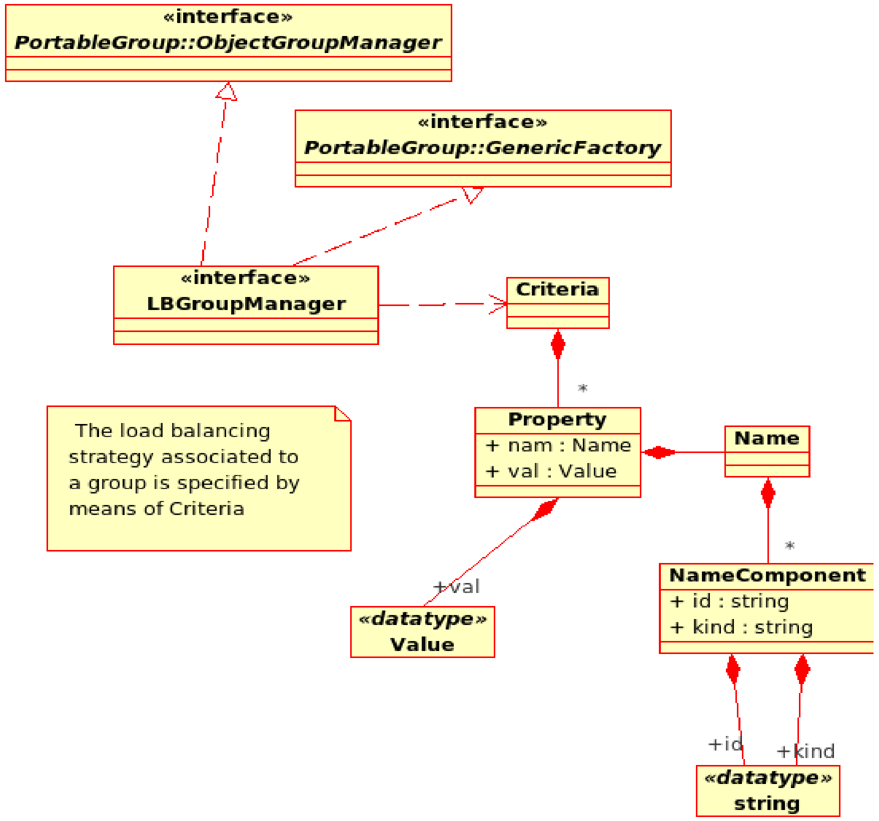

Figure 8.2 - LB Group Management

Finally, it is worth mentioning that, in order to take advantage of the load balancing, application shall invoke requests to the set of load balanced replicas by means of the ObjectGroup reference returned by the LBGroupManager.

8.3 Load Balancing Service

The LoadBalancingService interface provides a façade for managing load balancing strategies, along with the creation of load balancing groups (see Figure 8.3).

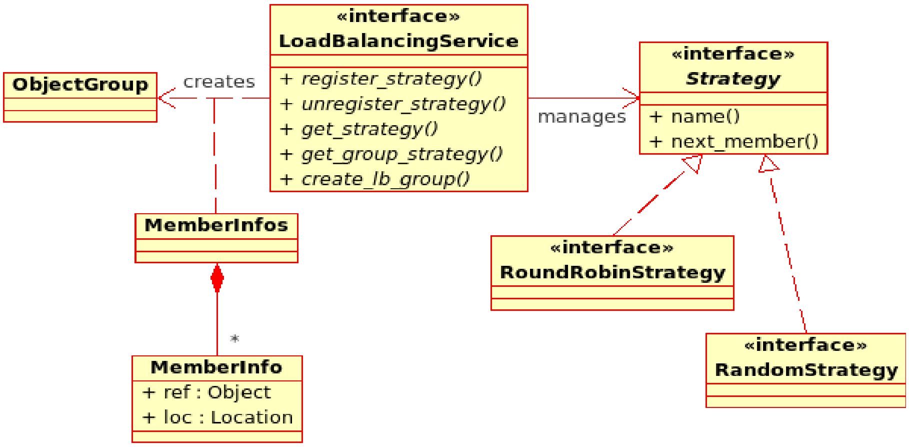

Figure 8.3 - Load Balancing Service

8.3.1 LoadBalancingService Class

This section provides a detailed description for the operations supported by the LoadBalancingService interface.

LoadBalancingService		
Operations		
1. create_lb_group	Creates a group of load balanced replicas. This operation encapsulates the facilities provided by the LBGroupManager.	
	Return Value:	
	PortableGroup::ObjectGroup	
	Parameters List:	
	members	MemberInfoSeq
	gid	PortableGroup::ObjectGroupId
	type_id	string
	lb_domain_id	string
	lb_strategy	string
	ref_version	PortableGroup::ObjectGroupRefVersion
	Exceptions:	
	PortableGroup::ObjectNotCreated	

2. get_group_strategy	Get the strategy associated with a load balanced group.	
	Return Value:	
	Strategy	
	Parameters List:	
	g	PortableGroup::ObjectGroup
	Exceptions:	
	InvalidObjectGroup	
3. get_strategy	Get the strategy object resolved by a given name.	
	Return Value:	
	Strategy	
	Parameters List:	
	name	string
	Exceptions:	
	UnkownStrategy	
4. register_strategy	Register a new load balancing strategy.	
	Return Value:	
	void	
	Parameters List:	
	strategy	Strategy
	Exceptions:	
	StrategyAlreadyRegistered	

5. unregister_strategy	Unregister a load balancing strategy.	
	Return Value:	
	void	
	Parameters List:	
	name	string
	Exceptions:	
	UnknownStrategy	

create_lb_group

This method provides a simplified API for the creation of load balanced groups—under the hood the implementation is delegated to the LBGroupManager. The object references for the group member along with their locations are specified by means of MemberInfoSeq.

get_group_strategy

This method returns the strategy associated with a group reference, or throws an exception if the group reference is not associated with a load balancing group.

get_strategy

This method resolve a strategy given its name. If the strategy name is not known, an exception is raised.

register_strategy

This method registers a new strategy with the load balancing service. This method can be used in order to provide user defined load balancing strategies.

unregister_strategy

This method removes a previously registered strategy. It is worth pointing out that any attempt to remove pre-defined load balancing strategies, such as ROUND_ROBIN and RANDOM will result in an LBGroupManager exception.

8.4 Load Balancing Strategy

The logic used to select replicas across client request invocation is controlled by means of strategy objects (see Figure 8.4). The Strategy interface, described below, provides a mean for controlling the selection logic, as well as plugging in user-defined behaviour.

8.4.1 Strategy Class

The following table provides a description for the operations supported by the LoadBalancingService interface.

Strategy		
Attributes		
	name	string
Operations		
next_member	Return the next server which will serve the request at hand.	
	Return Value:	
	Object	
	Parameters List:	
	object_group	PortableGroup::ObjectGroup

name

The readonly attribute name, provides a mean for accessing the load balancing strategy’s name.

next_member

This method actually executes the load balancing logic encapsulated by the strategy so to provide the next target replica to serve the client request at hand.

8.4.2 Strategy Invocation

This specification requires that the strategy is called at every client request performed on a group of load balancing nodes. However, it does not prescribe the mechanism to be used to invoke the Strategy while performing a call to a server object. Some implementation might rely on mechanism such as interceptors, while other might rely on other facilities.

The requirement set by this specification is that, however the strategy is called, an ObjectGroup, previously created by using the load balancing service, is provided to the strategy. As an example for the reader, Figure 8.4 provides an example in which interceptors are used in order to implement the service.

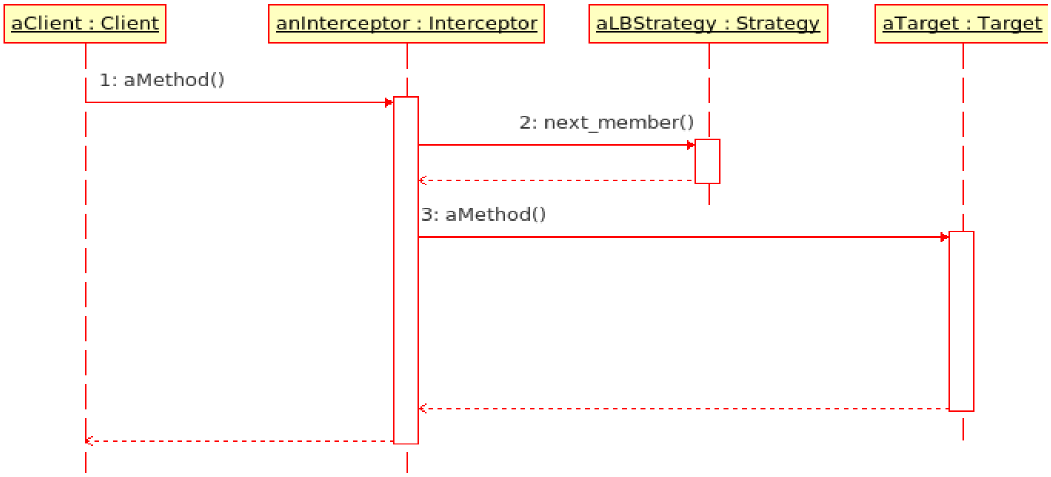

Figure 8.4 - Load Balanced Request

9 Platform Specific Model (PSM)

9.1 Mapping Rules

This specification currently foresees only a CORBA-based PSM. Thus the mapping rule are rather straightforward as most of the types described in the PIM have a direct IDL mapping. The consolidated IDL describing the CORBA PSM is provided in the following section.

Concerning with bootstrapping, this specification requires that the LBGroupManager as well as the LoadBalancingService interface are made available by means of resolve_initial_references. To this end, the ObjectId "LBGroupManager" is reserved for bootstrapping the LBGroupManager object, and the "LoadBalancingService" ObjectId is reserved for bootstrapping the LoadBalancingService object.

9.2 Consolidated IDL

```
#ifndef _OMG_ORG_COS_LB_
#define _OMG_ORG_COS_LB_

#include <PortableGroup.idl>

#pragma prefix "omg.org"


module CosLB {

  struct MemberInfo {
    Object        the_reference;
    PortableGroup::Location the_location;
  };

  typedef sequence<MemberInfo> MemberInfoSeq;

  interface LBGroupManager :
      PortableGroup::GenericFactory,
      PortableGroup::ObjectGroupManager
  {

  };

  interface Strategy
  {
    readonly attribute string name;
    Object next_member(in PortableGroup::ObjectGroup object_group);
  };

  exception UnknownStrategy {};
  exception StrategyAlreadyRegistered {};
  exception InvalidObjectGroup {};


  interface LoadBalancingService {
      void register_strategy(in Strategy s)
         raises(StrategyAlreadyRegistered);

      void unregister_strategy(in string name)
         raises(UnknownStrategy);

     Strategy get_strategy(in string name)
         raises(UnknownStrategy);

     Strategy get_group_strategy(in PortableGroup::ObjectGroup og)
         raises(InvalidObjectGroup);

     PortableGroup::ObjectGroup
     create_lb_group(in MemberInfoSeq members,
                     in PortableGroup::ObjectGroupId object_group_id,
                     in string type_id,
                     in string lb_domain_id,
                     in string lb_policy,
                     inout PortableGroup::ObjectGroupRefVersion ogrv)
        raises(PortableGroup::ObjectNotCreated);
   };

};
#endif /* _OMG_ORG_COS_LB_ */
```