维普资讯 http://www.cqvip.com

一种使用OCI的容错CORBA体系结构*)

黄小炜 赵致琢 吴文鑫
(厦门大学计算机科学系 厦门361005)

摘 要 本文提出了一种使用OCI的容错CORBA的体系结构。这里,我们使用的OCI在大部分的CORBA ORB中都有提供。我们替换OCI的网络协议,使用组通信系统来实现到对象组的通信。在我们的方法中,不需要对CORBA ORB和应用进行任何修改,或者只需要对应用进行少量的修改。

关键词 容错CORBA,开放的通信接口(OCI),组通信系统

A Fault Tolerant CORBA Infrastruvtion Using OCI

HUANG Xiao-Wei ZHAO Zhi-Zuo WU Wen-Xin
(Department of Computer Science, Xiamen University, Xiamen 361005)

Abstract In this paper, a fault tolerant CORBA infrastructure using OCI is presented, Where the OCI can be found in most CORBA ORBs. We substitute the network protocol of OCI with Group Communication System (GCS) to implement the communication to the object group. Our approach does not require modification to the CORBA ORBs and the application, or just require minimal modification to the application.

Keywords Fault tolerant CORBA, Open communication interface (OCI), Group communication system

1 引言

CORBA(Common Object Request Broker Architecture)是由OMG(Object Management Group)定义的一种分布式对象计算的中间件,它允许客户机调用远程对象上的操作,而不管该对象在何处运行以及用何种程序设计语言实现。由于CORBA的语言透明,位置透明,可互操作以及可移植性等特性,它极大地简化了分布式应用的开发,越来越多地为许多分布式系统的开发所采用。但是,随着CORBA应用领域的扩展,很多使用CORBA开发的应用系统都需要增加容错功能,所以OMG在2000年给出了容错CORBA(FT CORBA)的规范,并将其写入CORBA规范[1]。

组通信服务是用于保证所有成员对象一致性,并提供可靠多播的一种服务机制,实现该服务的系统称为组通信系统(GCS)。组通信系统维护一个视图(view),也就是当前在对象组中活动的并且连接的成员的列表,并当视图改变时通知运行的对象。组通信系统被广泛认为是构造容错应用的一种重要的技术。

开放的通信接口(Open Communication Interface,OCI)提供一个对网络协议的一致接口。它允许通过对OCI的实现很容易地在ORB(Object Request Broker)中插入新的协议或者其他的通信机制。OCI将消息协议(Messaging Protocol)和网络协议(Network Protocol)与ORB核心的其他部分以及应用对象分开。它允许开发者替换消息协议或者网络协议。

在容错CORBA规范中,最重要的概念就是对象复制。容错就是通过一个集合的对象副本(称为对象组)的协同操作取得的。到目前为止,各不同的厂商和研究机构已经提出或开发出一些与容错CORBA兼容的系统。一般说来,他们所采用的方法可以分成三类:(1)集成法[2,3],(2)服务法[4~6],(3)拦截法[7~9]。

在集成方法中,容错机制在ORB内部实现,是与ORB集成在一起的,所以该方法需要对ORB核心以及应用进行比较大的改动。在服务方法中,容错能力是通过在ORB之上的服务对象提供的。因为这种方法只能采用CORBA的点对点通信,所以无法在其中引入组通信机制。在拦截法中,容错机制的实现在ORB之下,与ORB有着不同的地址空间。在该方法中,可以通过拦截与组通信相关的系统调用来引入组通信机制。但是,因为拦截器本身不是CORBA的一部分,所以CORBA对象无法直接使用基本的组通信服务。

基于以上分析的目前容错CORBA规范实现方法的不足,我们考虑一种实现方法,它不仅可以与容错规范相兼容,而且可以引入组通信系统。而为了保证该系统具有可移植性,我们必须使得该体系结构对于任意的组通信系统都适用。而且,该方法应尽量避免对ORB核心和应用进行修改。

在本文中,我们提出了一种新的与容错CORBA规范兼容的容错体系结构,它是基于OCI的,并且适用于一般的组通信系统。其中,我们替换了CORBA的网络协议,而非消息协议。

2 The Open Communication Interface (OCI)

OCI为CORBA提供插件协议接口。在ORBacus中,这些接口有Buffer,Plugin,Acceptor,Acceptor Factory,Transport,Connector Factory,Registries和Info对象。Buffer以octet数组方式储存数据并维护一个位置计数器,该计数器用于确定多少octets已经被发送或接收。Plugin被用于初始化客户机和服务器的OCI插件。Info对象提供关于Transport,Acceptor,Connector,Acceptor Factory和Connector Factory的信息。

*)福建省自然科学基金项目,资助号:A0310007。

维普资讯 http://www.cqvip.com

图1说明 OCI 如何将一个给定的传输协议与 ORB 结合在一起。下面是 OCI 的执行过程：

1）当一个客户机和一个服务器被激活时，对象适配器(OA)中的 Acceptor Registry 创建一个 Acceptor Factory，而 ORB 中的 Connector Registry 创建一个 Connector Factory；

2）Acceptor Factory 激活一个 Acceptor；

3）Acceptor 创建 profile 信息，并且 OA 创建一个使用该信息的 IOR；

4）Connector Factory 创建一个使用该 IOR 的 Connector；

5）Connector 和 Acceptor 实例化它们自己的 Transport，然后客户机和服务器可以经由该 Transport 进行通信。

ORBacus 为接口 Connector Factory，Connector，Transport，Acceptor Factory 和 Acceptor 提供抽象的基类。协议插件必须从这些基类中通过继承这些类来提供对某一特定协议的具体实现。ORBacus 已经为接口 Buffer、Connector Factory Register 和 Acceptor Factory Registry 提供了具体的类。Connector Factory Registry 和 Acceptor Factory Registry 的实例可以通过分别使用标识"OCIConFactoryRegistry"和"OCIAccFactoryRegistry"，经由 ORB 操作 Resolve_Initial_Reference 得到。Connector Factory 的具体实现必须向 Connector Factory Registry 登记，而 Acceptor Factory 的具体实现也必须向 Acceptor Factory Registry 登记[11]。

OCI 接口的引入，使得我们可以简单地替换消息协议或者网络协议。D. Nam 等[10]通过对 OCI 的扩展，通过替换网络协议引入了一般的组通信系统。而下面将要介绍的体系结构也是通过替换网络协议得到的，但是，它并不对 OCI 进行扩展，而是利用 OCI 的接口在 OCI 和组通信系统之间引入容错机制。

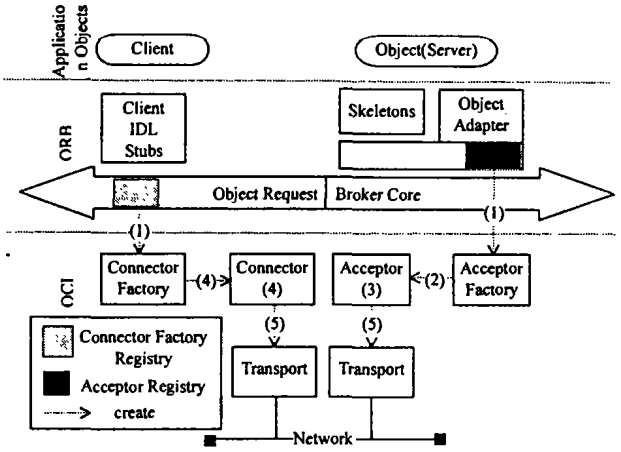

图1 Open Communication Interface(OCI)

3 一种新的容错 CORBA 体系结构

本文提出一种新的容错 CORBA 体系结构，如图2所示。在 server 端，我们用组通信系统(GCS)、容错协议插件、故障通知器、故障监测器和复制管理器来实现服务器的容错能力。其中，复制管理器、故障监测器和故障通知器被分别实现为一个 CORBA 对象，运行在与 server 不同的进程上，并采用主动复制方案进行复制，它们的实现完全遵照了容错 CORBA 规范。

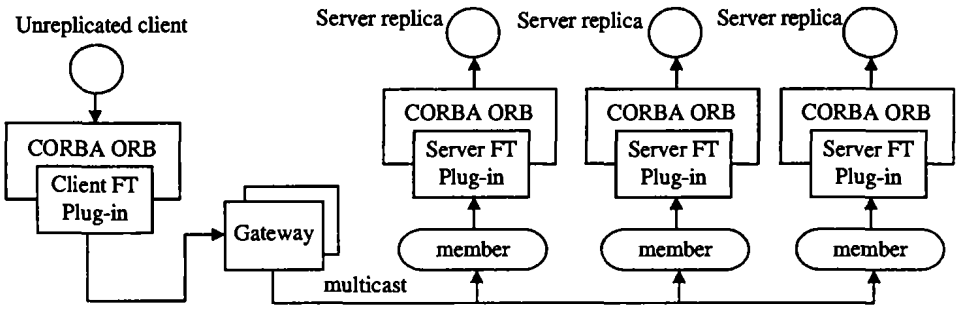

图2 The Proposed FT CORBA Infrastructure

在我们提出的体系结构中，一种组通信系统满足如下条件就可以被用做基本的通信方式：

1）必须向应用提供接口用于管理组成员，比如说 join，remove 等；

2）当视图改变时必须通知应用；

3）必须提供应用 broadcast 和 deliver 消息的接口；

4）必须提供可靠的组通信能力。

而我们目前看到的所有的组通信系统都满足这前四个条件，所以说，我们提出的这个体系结构对组通信系统具有一般性。

下面我们将讨论组成该体系结构的几个主要的成分。

1）*服务端的容错插件* 该服务端的容端插件是通过 OCI 插入到 ORB 中的。如图3所示，服务端容错插件是由以下组件构成的：组通信适配器，日志管理器，状态管理器和对象层故障监测器。

2）*组通信适配器* 其用于连接组通信系统和 OCI 的 Transport 实例，将输入的请求消息传给 Transport 实例，并将输出的回应消息路由交给组通信系统以便于发往目标对象或对象组。除了用于通信外，组通信适配器还须将输入的请求消息、输出的回应消息以及收到的 Primary 对象的状态写入日志管理器，将收到的视图改变消息传给对象层故障监测器。所谓 Primary 对象是指对象组中的主副本。此外，当复制管理器通知该副本为新的 Primary 对象时，还须将此信息写入日志管理器，便于状态管理器查询。

3）*日志管理器* 它记录最近一次 Primary 对象的状态，以及在该状态变迁之后的请求消息和回应消息的队列。同时，日志管理器也保存一些所在组的信息，如所在组的当前视图以及该副本是否是 Primary 等。当收到最近一次 Primary 对象的状态，日志管理器删除在该状态变迁之前的所有请求和回应消息，并替换储存在该组件中的 Primary 对象的状态。

4）*状态管理器* 实现一个接口，该接口提供一个方法允许应用登记它的工厂对象。这个工厂对象实现 FT::GenericFactory 接口。所以，状态管理器提供了容错 CORBA 中的 Factory 的功能，用于创建一个新的对象副本。此外，状态管理

维普资讯 http://www.cqvip.com

器接口提供 get_state () 方法来得到 Primary 对象的状态，并在必要的情况下调用应用对象的 FT::Checkpointable::set_state () 方法将 Primary 对象的状态赋给该对象副本。在恢复过程中，有如下三种情况：

(1)当启动新副本时，状态管理器使用登记的工厂对象创建该对象副本，并调用 get_state()方法得到 Primary 对象的状态，然后调用该副本的 FT::Checkpointable::set_state()方法设置该副本的状态；

(2)在 COLD_PASSIVE 复制体系下，当 Primary 对象发生故障时，复制管理器任命该对象副本为新的 Primary 时，状态管理器调用该副本的 FT::Checkpointable::set_state () 方法将该对象副本的状态设为日志管理器中保存的原 Primary 对象的状态，并重播日志管理器中保存的输入和输出消息；

(3)在 WARM_PASSIVE 复制体系下，当 Primary 对象发生故障时，复制管理器只需要将日志管理器中保存的输入和输出消息重播即可。

状态管理器接口的 get_state ()方法可如下实现：当 get_state ()方法被调用时，状态管理器向日志管理器查询本地副本是否 Primary，如果是，则调用应用对象的 FT::Checkpointable::get_state ()方法来得到 Primary 对象的状态，如果不是，则在该对象组内广播 get_state () 消息，当 Primary 对象收到该消息，调用自身状态管理器的 get_state () 方法，得到应用对象状态并广播状态消息，则组内对象(包括 get_state ()方法调用者)都可以得到 Primary 的状态。

5)对象层故障监测器的实现为一个 C++ 对象　被监视的 CORBA 应用对象副本必须继承 FT::PullMonitorable 接口。对象层故障监测器基于用户指定的故障监视时间间隔，周期性地调用 FT::PullMonitorable::is_alive ()方法来监视该对象副本。如果故障检测器发现该对象副本发生故障，则向故障通知器报告错误。此外，当对象层故障监测器收到视图改变信息时，与原先所存的组视图信息做比较，如果只是出现新的组成员，则将用该信息替换原先所存的信息，如果发现有成员缺席，则向故障通知器报告该成员发生故障。

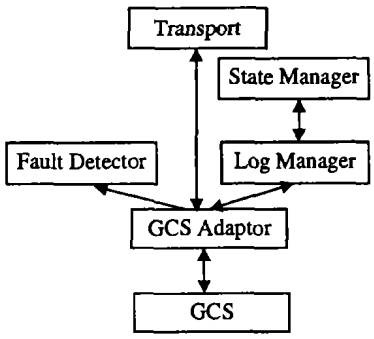

图3　Server-Side FT Plug-in

6)未复制的客户端容错插件　如图4所示。容错适配器使用 TCP/IP 来实现与容错域的主网关的连接和通信。容错适配器发送请求消息并接收回应消息，同时，它还要转发网关监视器对主网关的心跳监视消息以及主网关的回应消息。

客户端应用必须在网关状态管理器中登记服务器的 IOGR，并使用该 IOGR 的一个 profile 经由容错适配器建立与主网关的连接，然后主网关接受该连接并将消息广播给目标服务器对象组。因为我们在这个客户端和服务端连接的过程中使用了组通信适配器，所以这个调用过程对 client 是透明的。

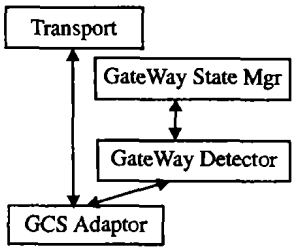

图4　client-Side FT Plug-in

当网关监视器确定当前建立连接的主网关已经崩溃，则命令组通信适配器关闭当前的连接并重新建立到新的主网关的连接。这里，新的主网关是在网关状态管理器中登记的 IOGR 中循环查找 profile 得到的。当所有的 profile 都被试过并且都已经崩溃了，则组通信适配器向客户机返回 exception。

4　体系结构的实现及其需要解决的问题

我们在 ORBacus4.0.4上进行该体系的实现。在实现中，复制管理器和故障通知器都实现为 CORBA 的服务对象，并且也对它们进行复制。而底层的服务器端的容错插件里的各个组件以及客户端的容错插件的各个组件都实现为 C++ 对象，并与 ORBacus 提供的 OCI 接口、组通信系统的接口以及其他内部成分之间进行交互。此外，还有些组件需要从复制管理器继承，如状态管理器，或者与故障通知器交互的，如故障检测器。

我们知道，在组通信环境下，由于需要在网络中广播消息，因此重复的消息是不可避免的。我们在实现时需要考虑如何消除这些重复消息。我们可以引入消息标签的方式，对每条消息以及它的回应消息进行标识。这样，对接收到的重复消息，我们可以很容易地识别。而且，因为重复的消息会对网络带宽造成一定的负面影响，所以最好的方法是在重复消息还没发送前就进行识别并消除。

在体系结构的描述中，为了简化，体系结构与 OCI 的内部组件的通信只是通过 Transport 组件，在实现时可以考虑将该体系结构与 OCI 内部组件进行结合，通过扩展 OCI 的方法来实现该体系结构。

5　对提出的体系结构的评价

下面给出几个对容错 CORBA 规范实现的系统的性能评价的标准，并对给出的系统进行评价：

1)对 ORB 核心的修改。该体系结构是通过 OCI 进入 ORB 的地址空间的，所以它不需要修改 ORB 核心。

2)对应用的修改。只需要 OCI 提供给用户可以一个选择特定容错协议的 API，并且如果 OCI 允许 ORB 在运行时装载特定协议的话，则根本不需要对应用进行修改。

3)可移植性。因为现在不同的 CORBA 系统的 ORB 对 OCI 的支持都遵循相似的面向对象设计模式，这就使得 OCI 方法具有较强的可移植性。

4)透明性。使用 OCI 的方法严格遵循了容错 CORBA 对透明性的要求。

5)可互操作性。在提出的结构中，我们在同一个对象组内必须使用相同的组通信系统，所以对可互操作性有一定的影响。

维普资讯 http://www.cqvip.com

参 考 文 献

1 Object Management Group (OMG). Common Object Request Broker Architecture (CORBA), v3.0. Dec. 2002

2 Cukier M, et al. AQuA: An adaptive architecture that provides dependable distributed objects. In: Proc. of the IEEE 17th Symposium on Reliable Distributed Systems, West Lafayette, IN, Oct. 1998. 245～253

3 Landis S, Maffies S. Building reliable distributed systems with CORBA. Theory and Practice of Object Systems, 1997, 3(1): 31～43

4 Felber P, Guerraoui R, Schiper A. The implementation of a CORBA object group service. Theory and Practice of Object Systems, 1998, 4(2): 93～105

5 Marchetti C, Mecella M, Virgillito A, Baldoni R. An interoperable replication logic for CORBA systems. In: Proc. of the International Symposium on Distributed Objects and Applications, Antwerp, Belgium, Sep. 2000. 7～16

6 Natarajan B, Gokhale A, Yajnik S, Schmidt D C. DOORS: Towards high-performance fault-tolerant CORBA. In: Proc. of the Intl. Symposium on Distributed Objects and Applications, Antwerp, Belgium, Sep. 2000. 39～48

7 Moser L E, Melliar-Smith P M, Narasimhan P. Consistent object replication in the Eternal system. Theory and Practice of Object Systems, 1998, 4(2): 81～92

8 Narasimhan P. Transparent Fault Tolerance for CORBA: [PhD thesis]. Department of Electrical and Computer Engineering, University of California, Santa Barbara, Dec. 1999

9 Narasimhan P, Moser L E, Melliar-Smith P M. Strong replica consistency for fault-tolerant CORBA applications. In: Proc. of the IEEE 6thWorkshop on Object-Oriented Real-Time Dependable Systems, Rome, Italy, Jan. 2001

10 Nam D, Lee D, Youn H Y, Yu C. Group communication support for CORBA using OCI. In: Proc. 12th IASTED Intl. Conf. on Parallel and Distributed Computing and Systems, Las Vegas, NV, Nov. 2000. 106～111

11 IONA Technologies, Inc. 2001. ORBACUS. http://www.orbacus.com/ob/.

12 Mena S, Schiper A, Wojciechowski P. A Step Towards a New Generation of Group Communication Systems: [Technical Report IC/2003/01]

(上接第102页)

标记会在当数据在客户 socket 上被写时插入到流出数据中。说明一点是，这个插入处理不会改变在贮存入口的数据内容。确保在代理服务器中被隐藏的数据拷贝不包含本地信息。

4 执行

我们通过实时的、基于代理服务器的 HTML 对象的传输来研究了我们为本地信息所设想的解决方案。依靠在更高的执行效果、全功能的代理服务器上的方案的执行，显示了方案的系统体系结构和设计的可行性和实用性。

第一个有影响的因素是相对 HTML 对象的访问延时的插入处理的运行时间。这个执行的测验是在基于 SQUID2.3 版本的代理服务器上进行的。它设置在一个双 CPU 的 PentuimIII1GHz，512M 内存，128M 交换空间和4GSCSI 硬盘计算机上。操作系统为 Linux RedHat6.0。所用的代理服务器是不含本地信息的相同的 SQUID 转置代理服务器，两者在同一台机器上测试。我们观察到对 HTML 对象的平均访问时间为1～2秒，在信息插入之上的额外的运行时间少于1毫秒，这是足够小以至可以被忽略的时间。

第二个有影响的因素是在 HTML 网页的访问延时之上的插入处理的运行时间。我们发现额外的延时由于下面二个原因事实上是不会被观察到的。

·每一个网页通常都包含多个嵌入的对象。由于信息标记的插入仅仅作用于 HTML 对象，它对整个网页恢复时间的作用效果远远小于 HTML 对象。另外，HTML 对象的访问延时往往比在网页中嵌入图像对象用时要小得多。

·在网页上的对象的恢复是运行在受限的并行模式下。因此，对一个对象访问的信息插入延时通常隐藏在数据传送通道中。

第三个影响因素是恢复信息条幅和呈现它给网页冲浪者的延时。这个延时依赖条幅图像的大小，条幅服务器的工作量以及在网页条幅服务器和网上冲浪者之间的可用的带宽。

结论 仿照电视广播里的插播模式，我们设想了这个动态的代理中心结构作为支持本地信息高效传送的手段，当网页通过与内容服务器协作的动态代理服务器被取回的时候，当地信息将基于需要灵活地被插入到网页中。这个网页信息通过代理服务器时在其中插入本地信息的代理服务器-中心的解决方案对于网页客户与内容提供商来说是完全透明的，切实可行的。

参 考 文 献

1 Abrams M, Standridge C R, Abdulla G, Williams S. Caching Proxies: Limitations and Potentials. In: Proc. of the 4th Intl. World Wide Web Conf. Dec. 1995. http://www.w3.org/Conferences/WWW4/Papers/155

2 Bennett D. How Interactive Ads are Delivered and Measurement Implication. White Paper, Audit Bureau of Circulation. http://www.accessabvs.com/webaudit/admeasurement.html.

3 Berthon P, Pitt L, Prendergast G. Visits, Hits, Caching, and Counting on the World Wide Web: Old Wine in New Bottles? Internet Research: Electronic Networking Applications and Policy, 1997, 7(1) http://www.emerald-library.com/brev/17207aa1.htm.

4 Broadvision Inc. http://www.broadvision.com

5 Cassidy P F. How New Auditing Tools Legitimize Web Advertising. Feb. 1997. http://www.netscapeworld.com/netscapeworld/nw-02-1997/nw-02-webaudit.html

6 Chi C-H, Li X. Content Transformation Model for Pervasive Internet Access: [Internal Report]. School of Computing, National University of Singapore, 2000. Also submitted to IEEE Multimedia and Expo, 2000