第34卷 第23期 计 算 机 工 程 2008年12月
Vol.34 No.23 Computer Engineering December 2008
·软件技术与数据库· 文章编号：1000—3428(2008)23—0071—03 文献标识码：A 中图分类号：N945

基于分布式对象的统一RTI接口设计与实现

刘 斌，张宏军，杨晓恕
(解放军理工大学工程兵工程学院，南京 210007)

摘 要：统一的RTI接口是为了解决异构联邦成员的互操作问题而提出的，其设计目标是具有广泛的兼容性和可扩展性。该文介绍了分布式对象中间件及其接口定义语言，建立了RTI的分布式对象模型和接口抽象，探讨了统一RTI接口在集成不同语言和不同协议的联邦成员中的应用以及在层次式RTI中的扩展。

关键词：分布式对象；接口定义语言；高层体系结构；联邦成员接口规范

Design and Realization of Unified RTI Interfaces Based on Distributed Objects

LIU Bin, ZHANG Hong-jun, YANG Xiao-jia
(Engineering Institute of Corps of Engineers, PLA Univ. of Sci. & Tech., Nanjing 210007)

【Abstract】Unified RTI interfaces, designed with extensive compatibility and scalability in mind, are proposed to solve the interoperability problem of the heterogeneous simulation federates. This paper briefly introduces the distributed objects middleware and the Interface Definition Language(IDL), on which the distributed objects model and abstract interface of the RTI are based. How to use unified RTI interfaces to integrate simulation federates in different languages or different protocols and how to expand it in the hierarchical RTI are discussed.

【Key words】 distributed objects; Interface Definition Language(IDL); High Level Architecture(HLA); federate interface specification

1 概述

基于高层体系结构(HLA)的分布式交互仿真是计算机技术的进步和仿真需求不断发展的结果，其特点包括分布性、交互性、异构性、时空一致性和开放性。在基于广域网的大规模分布式仿真系统中，异构性主要表现在：网络，操作系统，编程语言和交互协议。当前，主流的RTI系统针对不同的联邦成员开发环境大多采用独立的版本，如Win32版本和Unix/Linux版本，C++版本和Java版本，HLA1.3版本和IEEE 1516版本等。版本众多不仅增加了仿真应用开发的复杂度，而且基于不同版本RTI的联邦成员之间往往不能直接互联而需要使用转换器或适配器。因此，设计一种平台无关、兼容多种编程语言和交互协议的统一RTI接口，使得异构联邦成员可以在一体化的仿真环境下实现互操作成为RTI系统开发的重要目标。

分布式对象技术是面向对象的设计思想在分布式系统中的扩展，通过数据的封装和抽象接口的定义使得应用程序对分布更加透明。同时，由于策略与实现机制的分离，因此系统具有更高的灵活性和可扩展性。分布式对象与RTI系统的结合为统一的RTI接口设计提供了有效的解决方案。

2 分布式对象中间件概述

分布式对象是一种常见的中间件技术，它把面向对象技术中对象的封装性和继承性应用到分布式系统中，通过建立对象之间统一的接口，使对象的方法调用和数据共享不再关心对象的物理位置、所驻留的操作系统及实现语言。随着分布式系统和面向对象技术的结合，产生了大量基于分布式对象模型的标准化和商业化产品，如OMG组织的CORBA、Microsoft公司的DCOM/COM+、Sun公司的Java RMI和Zeroc公司的Ice等。

2.1 分布式对象模型

分布式对象的组织结构如图1所示[1]。

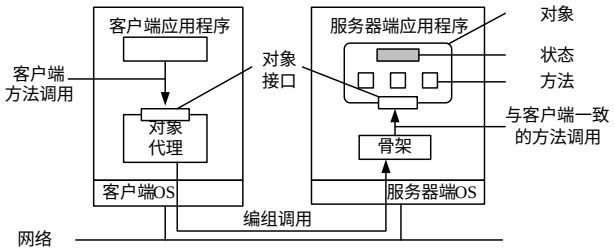

图1 分布式对象的组织结构

对象的关键技术是对数据的封装，这些被封装的数据称作状态，对这些数据的操作称作方法，只有通过接口才能使用方法。一个对象可以实现多个接口，同样的，在给定一个接口定义的情况下，也可以有多个对象提供该接口的实现。由于接口与实现这些接口的对象严格分离，因此可以把接口放在一台机器上，而让对象本身驻留在另一台机器上。当一个客户端绑定到一个分布式对象的时候，对象接口的一种实

基金项目：国家自然科学基金资助项目“合同战术训练评估系统”(70371039)

作者简介：刘 斌(1977－)，男，讲师、硕士研究生，主研方向：作战模拟与决策分析，分布式交互仿真；张宏军，教授、博士生导师；杨晓恕，博士研究生

收稿日期：2008-03-08 **E-mail：**l.bin.2008@gmail.com

现被加载到客户端的地址空间，通过该接口可以调用对象的方法并得到返回结果。对象在客户端接口通常被称作对象代理。在对象的实际驻留位置，也就是服务器端，对象提供与客户端一致的接口。输入的请求首先被传递给服务器存根——通常被称为骨架(skeleton)，它将调用请求进行编组，从而转换成对服务器上对象接口的对应方法调用。

2.2 接口定义语言(IDL)

分布式对象在客户端和服务器端之间的数据交互通常采用远程方法调用(RMI)。RMI是本地方法调用的自然扩展，它使得对象在网络上的分布对使用者来说完全透明。分布式对象通过接口定义语言(IDL)提供对RMI的支持。IDL是分布式对象接口的基本抽象，它为RMI提供了定义模块、接口、类型、属性和方法等机制[2]。不同的分布式对象中间件采用的IDL语法规则不尽相同，如CORBA的CORBA IDL、DCOM的MIDL、Web Service的WSDL和Ice的Slice，但它们都具备如下共同特征：

(1)类型系统定义：不论是分布式对象的状态，还是在RMI 中的传递参数，都需要一个抽象的类型系统加以描述，以确保在异构的环境下数据格式可以被正确解析。IDL 内置有部分简单的数据类型，同时提供自定义类型机制以构造较复杂的数据结构。

(2)对象接口定义：分布式对象中包含的方法必须通过IDL 定义后才可以被对象代理访问。对象接口的定义体现了分布式对象以接口为中心的设计思想。

3 统一的 RTI 接口定义

3.1 RTI 的分布式对象模型

作为HLA的运行支撑环境，RTI提供了一组服务和回调函数用于联邦执行生命期内联邦成员的互操作和数据交换，即联邦成员接口规范。该规范定义了联邦管理、声明管理、对象管理、时间管理、所有权管理和数据分发管理 6 大类服务及其相应的回调函数。RTI服务被封装在RTI大使类中，而回调函数被封装在联邦成员大使类中[3]，这 2 个类及其包含的方法构成了RTI接口的关键。RTI的核心组件包括本地RTI组件(LRC)和中央RTI组件(CRC)2 部分，联邦成员通过LRC的API接口调用RTI服务，在CRC完成相应的功能后把返回值或异常通过LRC传递给联邦成员。当需要通知联邦成员执行回调函数的时候，由CRC发起调用LRC的回调函数接口，并最终通知联邦成员。基于分布式对象的RTI接口被映射为 2 个分布式对象，分别命名为IRTIamb和IfedAmb，IRTIamb的实现在CRC完成，IFEDamb的实现在LRC完成。在LRC构造IRTIamb对象的代理即可实现对RTI服务的调用，同样在CRC中通过构造IFedAmb对象的代理即可实现RTI的回调函数。RTI接口的分布式对象模型如图 2 所示。

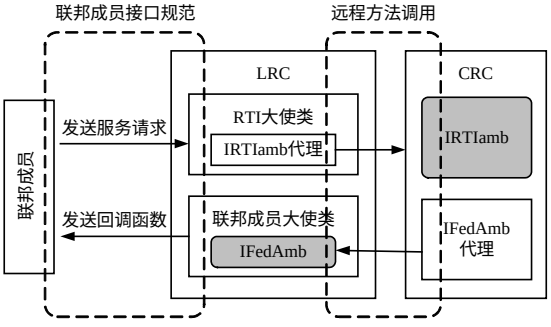

图 2 RTI 接口的分布式对象模型

3.2 基于 IDL 的 RTI 接口抽象

建立RTI接口的分布式对象模型后，还需要把相应的RTI服务及其依赖的特定数据结构用IDL加以描述。以Ice的 IDL Slice为例[4]，RTI的数据类型和服务接口原型实现如下所示：

```
// 类型系统定义
dictionary<long, Values> HandleValueMap; // “句柄-值”对集
sequence<long> HandleSet; // 句柄集
… // 其他 RTI 系统的抽象类型定义
// 异常定义
exception RTIexception {
  long exSerial;
  string exReason;
  string exName;}; // RTI 异常类的基类定义
exception ArrayIndexOutOfBounds extends RTIexception {...};
… // 其他 RTI 异常类定义
// 接口定义
interface IRTIamb {
  void createFederationExecution(string executionName, string FDD)
        throws RTIexception; // RTI 服务接口定义
  …}; // RTI 大使接口类定义
class IFedAmb {
  void synchronizationPointRegistrationSucceeded (string theLabel)
        throws FederateInternalError; // RTI 回调函数接口定义
  …}; // 联邦成员大使接口类定义
```

4 实现异构联邦成员的互操作

RTI 接口的开放性决定了系统能以各种不同的方式扩展和重用。基于分布式对象中间件的 RTI 接口具有平台无关、语言无关和实现无关的特性，是保证其有效屏蔽联邦成员异构性的关键。

4.1 连接不同语言的联邦成员

最常见的异构性是不同的联邦成员采用不同的程序设计语言开发。由于分布式对象中间件采用抽象的类型系统描述和统一的接口定义，使得基于不用语言开发的联邦成员之间的数据交互成为可能[5]。基于IDL的RTI接口定义可以转换为二进制接口或特定语言接口，前者是独立于编程语言的，后者则需要建立IDL与编程语言的映射，这个工作是由IDL编译器完成的。根据HLA规则，联邦成员之间的数据交互必须通过RTI而不能直接进行通信，因此，实现不同语言联邦成员的互操作实际上就是实现RTI与不同语言的联邦成员的互操作。以C++和Java为例，假设CRC是由C++语言实现的，而联邦成员由Java语言实现，此时RTI接口的语言映射关系如图3所示。

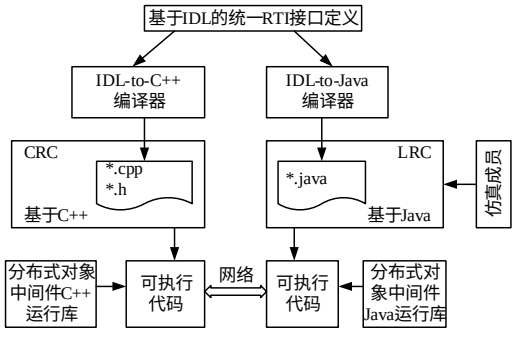

图 3 RTI 接口的多语言映射

4.2 兼容 HLA 1.3 和 IEEE 1516

HLA 规范主要由 3 部分组成：HLA 规则，HLA 对象模

型模板和 HLA 接口规范。

HLA规则在HLA 1.3 和IEEE 1516 之间没有区别，HLA 对象模型模板生成的联邦执行数据文件由RTI直接解析，因此兼容基于HLA 1.3 和IEEE 1516 的联邦成员关键在于提供兼容两者的交互接口[6]。

从接口的描述形式上看，IEEE 1516.1是HLA 1.3的超集，即 HLA 1.3 中定义的 RTI 服务和回调函数都可以在 IEEE 1516.1 中找到对应的服务和回调函数(除本地服务 tick 外)。

从接口的类型系统上看，IEEE 1516.1 中定义的数据类型与 HLA 1.3 也是基本一致的，利用 IDL 的类型系统可以很容易实现两者的抽象，如 3.2 节给出的 RTI 接口类型定义中，HandleValueMap 类型既可以作为 HLA 1.3 中的 AttributeHandleValuePairSet 和 ParameterHandleValuePairSet 类型的原型，也可以作为 IEEE 1516.1 中的 AttributeHandleValueMap 和 ParameterHandleValueMap 类型的原型。

不同版本的 LRC 把接口定义中的数据类型按照其标准 API 的形式进行封装，从而为联邦成员提供一致的访问接口。

4.3 在层次式 RTI 中的扩展

基于分布式对象中间件的 RTI 接口还可以扩展到基于互操作协议的层次式 RTI 系统中。

为了应对广域网的大规模分布式仿真，具有层次结构的联邦架构逐渐成为主流。

在一个层次式 RTI 中，若干个联邦成员可以组成一个子联邦，由本地 RTI 服务器(LRTI)负责联邦成员之间的交互。

子联邦可以组成更高级的联邦，由中央RTI服务器(CRTI)负责连接若干个子联邦[7]。

RTI 的层次结构对联邦成员是透明的，联邦成员只与管理它的本地 RTI 进行通信，本组内的联邦成员之间的数据交互由本地 RTI 处理并完成，不同组内的联邦成员之间的数据交互通过管理它们的局部 RTI 之间的通信完成，其交互流程如图 4 所示。当联邦成员的服务请求需要通过 CRTI 与异地联邦成员进行交互时，由 LRTI 向 CRTI 发送相应的服务请求，此时 LRTI 在 CRTI 中扮演着联邦成员的角色，因此图 4 中的 S1 与 S3、R1 与 R3、C1 与 C3 具有一致性，实现联邦成员与 LRTI 之间以及 LRTI 与 CRTI 之间的数据交互可以使用统一的 RTI 接口定义。

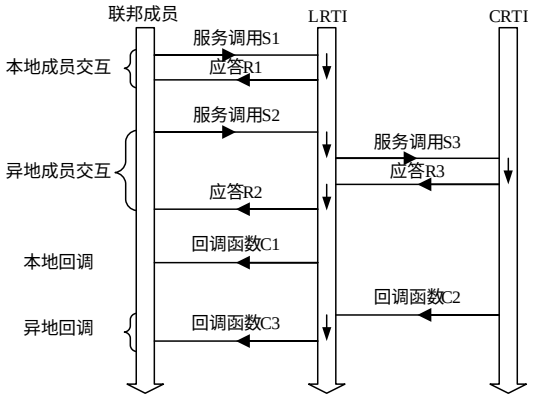

图 4 层次式 RTI 的数据交互流程

5 结束语

本文基于分布式对象构建的统一 RTI 接口定义，使 RTI 可以有效地屏蔽联邦成员的异构性，提高开发效率。文中所述的设计思想和方法已经被应用到 GY-RTI 系统的开发中，该系统是解放军理工大学工程兵工程学院信息技术系自主研发的新一代 RTI 产品，其设计目标是应对基于广域网的大规模训练模拟和综合仿真，经检验较好地满足了系统的设计需求。

参考文献

[1] Tannenbaum A S, Maarten van S. 分布式系统原理与范型[M]. 北京: 清华大学出版社, 2004.

[2] Coulouris G, Dollimore J, Kindberg T. 分布式系统概念与设计[M]. 3 版. 北京: 机械工业出版社, 2004.

[3] Simulation Interoperability Standards Committee of the IEEE Computer Society. IEEE Std 1516/1516.1-2000 IEEE Standard for Modeling and Simulation(M&S)[S]. 2000.

[4] Henning M, Spruiell M. Distributed Programming with Ice[EB/OL]. (2007-06-02). http://www.zeroc.com/.

[5] 韩 超, 鞠儒生, 黄柯棣. 基于 Web 服务的 HLA 仿真系统扩展[J]. 计算机工程, 2006, 32(24): 20-22.

[6] Möller B, Olsson L. Practical Experiences from HLA 1.3 to HLA IEEE 1516 Interoperability[C]//Proc. of 2004 Spring SIW Conference. Virginia, USA: [s. n.], 2004.

[7] 姚益平, 卢锡城. 层次式 RTI 服务器的设计与实现[J]. 计算机学报, 2003, 26(6): 716-721.

（上接第 37 页）

5 结束语

该系统已经初步设计实现，并交付金融自助软件应用程序开发实际应用，取得了较好的效果。实践证明用软件来模拟 ATM 机硬件系统的操作是完全可行的，并且有着十分积极的意义。但是任何行业都存在兼容性的问题，不同的硬件厂商的产品有着巨大的差异。目前在金融自助产品生产厂商中间该矛盾也很明显，用相同的软件模拟不同厂商的设备时，仅仅通过修改一些配置文件而不去修改程序，这对软件本身的架构提出了更高的要求，而且各个硬件厂商提供的上层接口的不一致性，也使得这种做法的可行性有了不确定的因素，因此，该设计有待进一步考证和研究。

参考文献

[1] 崔佳山. 银行自动柜员机业务系统的设计与实现[D]. 长春: 吉林大学, 2000-05.

[2] Nixdorf W. Protopas Programming Guide V3.1[Z]. 2005-03.

[3] 阎 宏. Java 与模式[M]. 北京: 电子工业出版社, 2002-10.

[4] 沃尔斯, 布雷登巴赫. Spring in Action[M]. 李 磊, 程 立, 周悦虹, 译. 北京: 人民邮电出版社, 2006.

[5] Lewis B, Berg D J. 深入学习: Java 多线程编程[M]. 北京: 电子工业出版社, 2000-12.