一个支持开放分布式处理的 DRPC 模型*

汲 化　谢 立　孙钟秀

(南京大学计算机系　南京　210093)

摘要　开放分布式处理 ODP(open distributed processing)的目标是试图解决分布环境下软件的接口问题,以达到分布式系统的可移植性、透明性、互操作性.本文介绍了 ODP 要达到上述目标所需解决的一些关键性问题,分析了传统的远程过程调用模型的服务静态连接机制和其对 ODP 支持的局限性,并提出一个改进的、支持服务动态连接的远程过程调用模型,可以获得较好的性能.

关键词　开放分布式处理,远程过程调用,静态连接,动态连接.

开放分布式处理 ODP(open distributed processing)是分布计算发展的方向,其试图解决分布环境下软件的接口问题,以达到分布式系统的可移植性、透明性、互操作性.[1]开放分布式处理参考模型 ISO RM_ODP 即将成为国际标准(ISO 10746 和 CCITT X.900 系列)[2],它定义了 ODP 的目标框架,但没有给出其具体实现环节.为了达到上述目标,目前尚存在一些关键问题需要解决[3~5],例如:(1)分布式系统的一个重要问题就是透明性,即使得用来克服分布问题和实现的机制相对用户应用无关,如存取透明性、位置透明性,它是分布式系统开放程度的一个重要标志.(2)在分布式系统中,构成系统的部件将由于系统扩张、改良、排障或负载平衡等原因而不断动态地变化,这就要求系统必须能够支持服务的动态连接,即顾客(Client)必须能在运行时刻发现哪一个服务员(Server)可提供所需的服务,它满足 Client 所要求的服务类型和特性,并动态连接到该服务所在的 Server 上.(3)构成系统的各部件之间必须具有互操作性,即有意义地交换信息和互相使用系统功能,因此在各部件之间必须有一个转换和管理调节机制来实现互操作.

本文分析了传统的远程过程调用模型的服务静态连接机制,讨论了其对 ODP 支持的局限性,并提出一个改进的、支持服务动态连接的 DRPC 模型,可以获得较好的性能.

1　远程过程调用

远程过程调用(RPC)在开发分布式 Client/Server 程序时,是一个非常有用的机制.RPC 的性质,类似于程序中的局部函数(Local Function)调用,所不同的是局部函数与调用

---

* 作者汲化,1969 年生,博士生,主要研究领域为分布式系统,并行处理.谢立,1942 年生,教授,博士导师,主要研究领域为分布式系统与并行处理.孙钟秀,1936 年生,教授,中国科学院院士,主要研究领域为分布式系统与并行处理.

本文通讯联系人:汲化,南京 210093,南京大学计算机系

本文 1995－09－14 收到修改稿

它的应用程序在同一台机器上,而 RPC 调用一般却是在另一台机器上执行完毕后,再将结果返回调用它的应用程序.RPC 交互模型提供一个高层接口,隐蔽分布的程序间通信的细节.这样分布式程序开发者不必关心网络间复杂的数据传送、接收、同步等问题,从而使得开发变得简单、方便.所以它是分布式系统中的一个重要部件.[6]

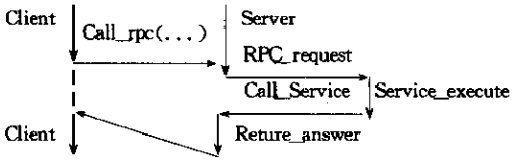

图1 SUN RPC调用模型

RPC 交互模型中,一个关键的问题是 Client 与 Server 的连接.由于 Client 与 Server 往往位于不同的机器上,所以连接也就是 Client 程序能正确寻址,定位含有所需 RPC 服务接口的 Server 的过程.目前的 RPC 模型中,大多数采用静态(Static)连接的方式,即在编译时刻 Client 必须事先确定 Server 的地址、RPC 接口的标识 ID,并且一个 Client 调用某个 RPC 过程时,与另一个特定的 Server 是严格的 1:1 关系,如 SUN RPC,用户利用 Call RPC 调用一个位于其它机器上的函数时,必须明确给出远程机器名或地址、标识 RPC 接口的 Server 程序号、版本号、过程号,方能建立与 Server 的连接,完成 RPC 调用.而这些信息的获取,RPC 交互系统并不提供,只能通过用户显式地获取.[7]如图 1 所示,为 SUN RPC 调用模型,是一典型采用静态服务连接的 RPC 调用模型.

这样的服务静态连接机制存在如下局限:(1) 缺乏灵活性和移植性.Client 必须事先获得连接 Server 所需的结点地址等信息,并显式地嵌入到 Client 程序中,当 Server 的地址改变时,Client 程序必须重新编译、生成,方能正确定位 Server,调用其中的 RPC 过程.(2) 缺乏分布透明性.评价一个分布式系统时,对服务的透明访问是一个重要的指标,而位置透明性则是透明性中的重要内容.静态连接方式必须显式地了解服务所在的位置信息,不能透明地访问服务.(3) 缺乏互操作性和可靠性.由于 Client 与 Server 是 1:1 的关系,所以当被连接的 Server 所在结点出现故障时,连接过程将失败.用户必须显式地面对失效和故障等错误,即使网络其它结点中存在能完成相同 RPC 服务的 Server 时,对于 Client 而言,不能透明地获得服务.

2 DRPC 模型

DRPC 是一个建立在图 1 基础之上、支持服务动态连接的 RPC 调用模型.动态连接意谓着 Client 在运行时刻发现、连接 Server 中的 RPC 接口.DRPC 认为网络中任一结点既可以是某一 RPC 过程调用方,同时也可能是另一个 RPC 接口的服务方.

2.1 概 念

首先,我们引入如下概念:

RPC_Signature :定义为一个 RPC 过程的函数名,输入参数类型和返回值类型.如果一个 RPC 过程调用为:$W(x1:\sigma1,x2:\sigma2,\cdots,xn:\sigma n)$

其返回值为:$(y1:\rho1,y2:\rho2,\cdots,ym:\rho m)$

则其 RPC_Signature 为:RPC_Signature :$=(W:\sigma1,\sigma2,\cdots,\sigma n:\rho1,\rho2,\cdots,\rho m)$

W 为函数名;$\sigma1,\sigma2,\cdots,\sigma n$ 为输入参数类型;$\rho1,\rho2,\cdots,\rho m$ 为输出参数类型.

系统中每个RPC过程都对应一个RPC_Signature.

RPC_ID:系统中每个RPC过程都存在一个ID(Identifier)与其对应.ID包括结点地址、程序号、版本号、过程号,可以表示为:

RPC_ID:=(Node_Addr;Prog_Num;Vers_Num;Proc_Num)

Client欲调用一个Server程序中的RPC过程,必须首先获得RPC_ID;每个RPC_ID必定与一个RPC_Signature相对应.

RPC_Interface:定义RPC_Interface为:

RPC_Interface :=(RPC_Signature;RPC_ID)

RPC_Interface完整地描述一个RPC过程实例的全部信息.如果2个RPC_Interface的RPC_Signature相同,而RPC_ID不相同,则这2个RPC_Interface代表完成相同RPC服务的2个过程实例.

RPC_Space:定义为一存储空间,保存着网络中现有的全部RPC接口的RPC_Interface信息.它由分布在各个网络结点上的本地RPC_Space共同构成,各结点维护自己的本地RPC_Space,本地RPC_Space保存着本结点上的RPC_Interface信息,各结点互相联合,构成全局RPC_Space.

RPC_Cache:每个结点上,除了拥有本地RPC_Space外,还定义一个一定大小的、保存着非本地的RPC_Interface信息,提供给Client查询,以提高系统效率,减少网络传输开销.

RPC_Manager:由分布在各结点上的RPC代理(RPC_Agent)共同构成.RPC_Agent管理本地RPC_Space,RPC_Cache,负责与Client或Server交互.RPC代理之间相互协调构成RPC_Manager.

2.2 DRPC模型

在上述概念基础上,可以定义DRPC的如下行为(Behaviour):

对于Client和Server,定义如下操作:

Export:一个拥有RPC过程接口的Server程序在初始化之后,向本地的RPC_Agent输出,通告其RPC_Interface,若成功则返回True(表明该服务准备好,可以被Client调用),否则返回False.其调用形式为:Export(RPC_Interface).

Import:一个试图连接调用某RPC过程的Client程序向本地RPC_Agent申请,RPC_Agent根据其提供的RPC_Signature,寻找其RPC_ID,若成功,则返回一个RPC_ID,否则,返回False,其调用形式为:Import(RPC_Signature).

对于RPC_Agent,如图2所示,可定义其行为如下:

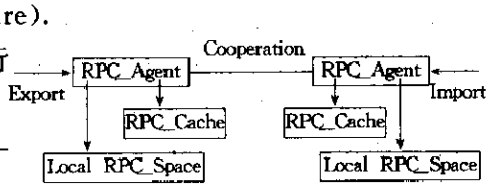

图2

当接收Export操作时,RPC_Agent将Server通告的RPC_Interface信息放入本地RPC_Space中,如果成功,则返回True,以表示此RPC服务准备好,可以接受远程Client的连接和调用,否则返回False.

当接收Import操作时,RPC_Agent首先在本地RPC_Cache中依据参数RPC_Signature查找,若成功,则返回匹配的RPC_Interface中的RPC_ID,否则,与网络中其它RPC_Agent协调合作(如利用Broadcast或Multicast),在它们的本地RPC_Space中查找并等待

返回结果，然后将返回值作为该次 Import 操作的结果返回给 Client. DRPC 基于如下知识：最先返回的 RPC_ID 所在结点负载较轻，与本结点之间传输数据延迟较小；同时 RPC_Agent 将这个 RPC 服务的 RPC_Interface 信息存入 RPC_Cache 中，并以进入 RPC_Cache 的先后来排序，以备下次查询. 如果 RPC_Cache 满，则删除在 RPC_Cache 中时间最长的记录，以用于存储最新的 RPC 信息.

对于 RPC_Space 和 RPC_Cache 的组织，可以采用按记录存取的方法，每个记录为一个 RPC_Interface，共包含 2 部分，即 RPC_Signature 和 RPC_ID. 其形式为：

```
Record:=
RPC_Interface
    RPC_Signature
    RPC_Signature
    END
    RPC_ID
    RPC_ID
    END
END
```

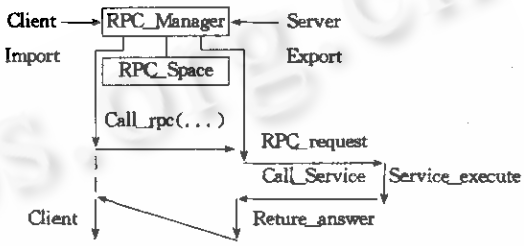

图3 DRPC模型

其中各 RPC_Signature，RPC_ID 之间，各字段之间都用明确定义的标识串和分隔符区分，以方便 RPC_Agent 的查询.

定义了上述行为之后，Client 程序就可以在运行时刻通过 Import 操作进行 RPC 服务的动态查询，发现所要调用的 RPC 过程的连接信息 RPC_ID，通过 RPC_ID，就可以实现与特定 Server 的连接，激活并完成 RPC 服务的调用（见图 3）.

3 DRPC 特点

我们在基于静态服务连接的 RPC 模型上定义了一些概念、部件和操作，得到一个支持服务动态连接的 RPC 模型，能较好地支持 ODP 对移植性、透明性、可靠性的要求，具有如下特点：(1)Client 程序在运行时刻发现、连接、调用所需的 RPC 过程，不必在编译时刻就确定 Server 所在地址等信息，所以当 Server 的地址改变时（如人为的改变或进程的动态迁移），Client 程序不需要重新编译，不必束缚于某一特定的结点上，则有较好的可移植性和灵活性. (2)因为服务被系统统一管理，能自动地定位服务，不需要用户或应用程序显式地了解资源地址、标识等信息，可以支持 ODP 系统中资源访问的位置透明性和系统配置动态性的要求. (3)Client 面对的不再是某特定的 Server，而是某种抽象的服务（RPC_Interface），突破了静态服务连接模型中 Client 与 Server 严格的 1：1 关系. 由于 Client 与 RPC_Interface 是 1:m (m≥1)，因此只要系统中存在所需的 RPC 过程，理论上就可以获得服务.

4 结束语

本文从开放分布式处理的角度出发，分析了基于静态服务连接的传统 RPC 模型的局限性，提出一个支持服务动态连接的 DRPC 模型，定义了其部件和行为，可以具有较好的位置

透明性、可移植性、互操作性和可靠性.

ODP还有很多问题要研究,该模型如何进化为开放分布式系统处理参考模型(ISO RM_ODP)中的资源媒介Trader将是我们的下一步研究方向.

参考文献

1 Herbert A. The challenge of ODP (open distributed processing). Elsevier Science Publishers B. V., IFIP, 1992.

2 ISO/IEC JTC/SC21. Basic reference model & open distributed processing. Part 1～4, 1993.

3 Nehmer Jurgen *et al*. Framework for the organization of cooperative services in distributed client－server systems. Computer Communications, 1992, **15(4)**.

4 Huang, Lamb D A. Communication abstract data type values in heterogeneous distributed programs. 14th ICDCS, Poland, 1994.

5 Tschammer V *et al*. Cooperative management in open distributed systems. Computer Communications, 1994, **17 (10)**.

6 Liu Yusheng, Hong Doan B. OSI RPC model and protocol. Computer Communication, 1994, **17(1)**.

7 Sun Microsystems. Networking on the Sun workstation.

8 汲化,吴迎红,周晓波,谢立.一个支持服务动态连接的远程调用模型的设计与实现.'95全国开放系统会议(论文集),1995.

A DRPC MODEL FOR OPEN DISTRIBUTED PROCESSING

Ji Hua　Xie Li　Sun Zhongxiu

(*Department of Computer Science · Nanjing University　Nanjing*　210093)

Abstract　The ODP (open distributed processing) aims to solve the software interface problems to achieve portability, interoperatability, and distribution transparency. This paper presents some of the existing problems for ODP to achieve its goal, analyses the service static binding mechanism of the traditional RPC model in the view of ODP. An improved, supporting service dynamic binding RPC model is proposed, through which better performance can be obtained.

Key words　ODP, RPC, static binding, dynamic binding.