·并行与分布式系统·

Client/ Server 计算的协作模型及其开发工具

黄海宁 孙伟伟 夏宽理 赵文耘 钱乐秋
(复旦大学计算机系 上海 200433)

摘要 该文提出了一个 Client / Server 计算的协作模型，可支持 Client 激活多个子服务以及多个子服务之间的协作交互，并在此基础上，设计了一个基于代理的 Client / Server 开发工具。

关键词 Client / Server 计算 协作模型 代理

Coordination Model and Development Tool of Client/ Server Computing

Huang Haining Sun Weiwei Xia Kuanli Zhao Wenyun Qian Leqiu
(Department of Computer, Fudan University Shanghai 200433)

【 Abstract 】 In this paper, the basic Client / Server Model are extended to the coordination model. In this model, the case of multiservers offering complex service to one client are discussed. Then an development tool are build to help to implement this model.

【 Key words 】 Client / Server computing; Coordination model; Agent

分布式计算系统是由多个分散的处理资源组成的计算系统。这些分散的资源对集中的程序、数据或硬件依赖性最小，在整个系统的控制下协作运行。*

1 Client / Server 协作模型

1.1 模型概述

所谓协作，在不同的上下文环境下有不同的意义，广义地说，是将各项相互关联的子任务集成地加以控制以完成更广泛和复杂的任务。具体地，它包括如何将任务分解成独立的或相关联的子任务，如何将子任务分配给各有关过程去执行，同时必须对协作执行进行控制，它还包括采用何种资源分布策略以及如何实现资源的共享等方面。

在基本的 Client / Server 模型下，如果一个客户使用多个服务器，他必须负责按自己的需求去使用它们，客户必须预先知道运行所需服务的服务器的地址，同时，服务器进程的协作也必须由客户机自己来控制，对于这种模型，当服务器增加或减少等情况引起资源分布的改变时，必须更改包括客户和服务器的几乎所有的程序，这种存取问题在异种系统中变得特别突出，因为请求服务的客户可能与不同的服务器和平台打交道。

为了保证灵活性和可用性，一个好的分布式系统，其分布式操作对用户应当是完全透明的，在 Client / Server 模型下，服务器的透明性是指服务器地址等细节对于客户的不可见性，让客户自己去控制服务器的协作执行是不透明的。

对于在一个客户与多服务器协作交互的情况下，我们使用服务器组来表示一组服务器，同组的服务器用于协作地完成一个功能集合，这里的服务器是一个逻辑概念，并不是物理上的服务器，单个服务器可以根据不同的运行状态可以属于几个不同的服务器组。使用服务器组的概念可以刻划单客户多服务器交互的模型，将不同服务之间的交互概括为服务器组内的交互，我们称之为组内交互。

客户与服务器之间的交互依赖于请求和响应的网络通讯模型，可能是阻塞的或非阻塞的方式。阻塞方式有利于客户进程和服务器进程的同步。而非阻塞通讯具有并行性，可实现客户进程和服务器进程的异步通讯。在 Client / Server 协作模型中，发出同步请求

* 黄海宁 男，25 岁，硕士生，主要研究领域：软件工程与环境、Client / Server 计算模型、数据库等

收稿日期：1996-12-17

的客户将阻塞，直到接收到服务器的响应。阻塞方式的实现比较简单，但是对于客户进程的处理能力来说是低效率的。非阻塞通讯方式使得客户在发出请求后能执行其他操作，从而提高了执行效率，但是也增加了为获得服务器响应的控制复杂性。

为解决 Client／Server 计算的协作问题，首先必须解决资源分布问题。资源包括数据资源和程序资源，我们引入对象的概念来描述分布于各个节点的数据资源，一个对象由特定的属性以及可以发生在该对象上的操作所组成，我们使用对象组来描述具有共同的状态特征的一组对象，运行在一个对象组上的所有操作进程构成了一个进程组，对该组对象进行操作的进程都属于该进程组，即所有引起该对象组状态的操作进程构成一个进程组。当进程组中的进程用于向客户提供服务时，我们称该进程为服务进程，进程组中的所有服务进程构成了服务器组。

1.2 协作管理器

在决定了资源分布策略以后，必须建立一个管理客户请求并响应请求的机制，我们定义一个协作管理器来接受所有来自于客户端的请求，由该协作管理器控制在不同的服务器上协作地执行请求，并将不同的服务器上获得的运行结果加以组合，然后由协作管理器将最终结果传送至客户端。

使用协作管理器的好处是：(1) 客户的代码独立于服务器组中服务器的数目，也即独立于 Client 与 Server 的布局结构，使得客户代码仅关心与最终用户的界面操作。(2) 服务器的代码独立于同服务器组中其他服务器的数目，也即使得服务器代码最大限度地独立于数据资源的分布及服务器组的划分，而仅仅关心其所提供的服务；(3) 用于提高性能和容错能力的代码被封装至协作管理器上，协作算法不属于客户和服务器代码的范围，因此，不必改动客户和服务器的代码就可以提高服务的性能和容错能力；(4) 同时提供一个基本的 API 界面供应用程序调用，简化客户程序和服务程序的编程。

1.2.1 请求代办者

协作管理器首先必须管理来自于客户的请求，我们定义一个请求代办者，它负责协调客户应用程序需要的服务和服务器应用程序所能提供的服务。请求代办者使客户不必关心在哪里和如何获得特定服务。所有分布于系统各节点的应用程序通过"请求代办者"注册它所能提供的服务和需要请求的客户接口，注册服务时必须提供服务的地址及接口。通常，请求代办者管理一个名字服务或目录。基于"请求代办者"模型，服务可以运行时动态注册或在编译时静态注册。请求代办者可以使用如下 3 种设计模型之一完成 Client／Server 交互。

完全代办模型：客户传递请求至代办者，代办者将客户请求传递至相关的服务器程序，并取回响应，最后将响应传递至请求客户。

句柄代办模型：客户传递请求至代办者，代办者处理该请求并返回一个句柄至客户，该句柄包含了与服务器交互的关于该服务的所有信息(包括名字、网络节点地址、接口等)，客户通过该句柄与服务器应用程序交互。

混合代办模型：同时支持以上两种模型，让客户在发出初始请求时用参数来选择。

完全代办模型提供了控制所有客户请求和服务器响应的场所，因此可以检测失败的交互并可重试，容错性和可恢复性不必由客户实现。缺点是消息传递的集中性有可能使性能降低。句柄代办模型减少了潜在的瓶颈问题，来自于服务器的响应避开了代办者而可以直接传递至客户。而混合代办模型则同时提供了两种机制供客户灵活地选择使用。

1.2.2 服务控制器

协作管理器中的核心部分是服务控制器，客户发出一个隐含服务请求，这个隐含服务请求不能由一个服务器直接完成，服务控制器必须将客户请求映射或分解成各个子服务请求并分析存在于各子服务之间的依赖关系，然后协调各个相关服务器的活动并收集响应，最后将这些响应组合成的一个单一和一致的响应并传递给客户。

如果各子服务相互独立，则可采用非阻塞模型，向有关的服务器发出请求并行执行，最后合并结果。相反地，对于相互依赖的服务，必须具有同步约束的机制，可采用阻塞的通讯模型。

1.2.3 服务器组管理器

我们定义了一个服务器组管理器作为协作管理器的一部分用于维护服务器组的静态和动态信息。

在分布式系统中，有两种管理资源的方式，一种是采用一个节点统一管理，称为集中分布管理方式；另一种是由多个节点共同管理，称为全分布管理方式。集中分布管理实现比较简单，而全分布管理的实现比较复杂，一般采用混合管理方式，即对一些资源采用全分布管理，而对另外一些资源实行集中分布式管理。

我们在对分布数据资源进行对象分析的基础上，建立资源分布策略，并为各个资源对象建立操作进程

组并提炼出服务进程构成服务器组。使用服务器组管理器来管理服务器组，由服务器组来管理分布的数据资源，既具有全分布管理的特点，又保留了集中管理的简洁性。

另外，考虑到服务器组的动态性，服务组管理器还必须提供动态创建、更改、取消、合并一个服务器组的功能。

1.3 服务器组协作计算开发工具

实现分布式 Client / Server 模型需要诸如远过程调用(RPC)和消息传递等工具以建立和控制网络上应用程序间的通讯，消息系统通常提供了一个发送消息至发送队列和从接收队列中接收响应的应用程序接口(API)，这种设计增加了应用程序的额外控制能力，导致了应用程序独立性降低，也影响了分布式系统的模块化、可维护性和可扩展性。为解决这个问题，通常在应用程序与网络 API 之间提供一个代理(agent)，这些代理提供了清楚的、结构化的构件以调度应用程序、通讯核心，以及执行应用集成或分布控制的其他任务。即代理提供了一个类似 RPC Stub 程序的中间角色的功能。代理可以用于实现基本的 Client / Server 模型。

基于 1.2 节所描述的协作模型，我们开发了一个协作计算开发工具，其核心包括了基于代理机制的协作管理器的协作模型应用程序接口。

协作管理器是一个逻辑概念，在实现方式上，既可以建立一个协作服务器节点，也可以分布于多个节点上。但是，如果在一个节点上实现，协作管理器容易成为系统的瓶颈，在分布式系统中，要尽量将协作管理器分布在多个节点上，但这种分布性势必增加控制复杂性。

将协作管理器建立在服务器组的基础上，即在系统运行的某一个状态，每个服务器组都对应一个协作管理器来管理和协作该服务器组上服务运行：同时，一个协作管理器可以管理一个或多个服务器组，协作管理器可以在或不在同一个服务器组所在的节点上(如图 1)。

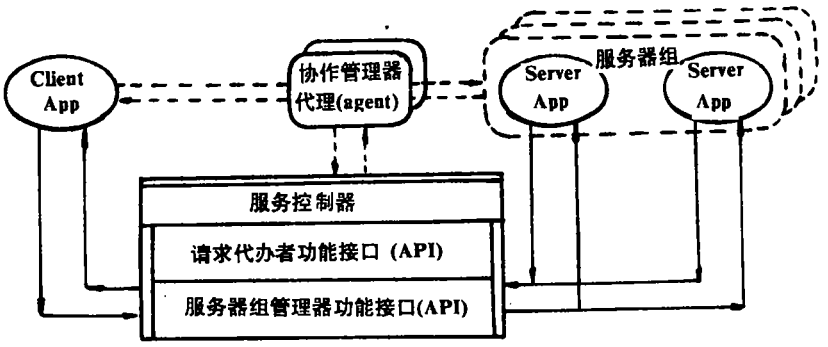

图 1　协作模型实现

基于上述观点，我们开发了一个协作计算开发工具(FD-CDT)，它包括了一个基于代理机制的协作管理器和协作模型应用程序接口。

请求代办者 API 主要函数：

(1) 基本函数：RBRegisterAPP　注册一个应用程序；RBRmvAPP　注销一个应用程序；RBRegisterService　注册应用程序所能提供的服务(服务名，服务地址，参数)；RBRmvService　注销某服务的注册(服务名)；RBRegisterRequest　注册应用程序需要的隐含服务请求(服务名，隐含服务描述)；RBRmvRequest　注销应用程序的隐含服务请求(服务名)。

(2) 为客户提供的函数：RBInvokeService　采用完全代办模式请求服务(服务名，参数)，返回结果；RBGetServiceHandle　采用句柄代办模式请求服务作准备(服务名)，返回服务句柄；RBRequestUseHandle　采用句柄代办模式请求服务(参数)。

(3) 为服务器提供的函数：RBReply　采用完全代办模式返回结果(服务名，结果)；RBReplyUseHandle　采用句柄代办模式返回结果(服务名，结果)。

服务器组 API 主要函数：SGCreate　创建一个服务器组(组名，服务名表)；SGDelFrom　从一个服务器组删除一个服务(组名，服务名)；SGAddTo

(下转第 59 页)

地确定其在每一细节层次上的分割特性。为此目的，建立以下的数据模型(如图1)：

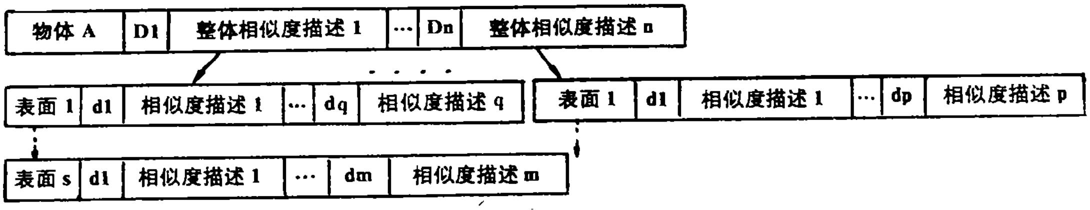

图1 数据模型

说明：

Di：代表物体整体离用户的距离。距离在 Di−1 到 Di 的，其相似度属于"整体相似度描述 i"。

di：代表视点和焦点的距离。距离在 di−1 到 di 的，其相似度属于"相似度描述 i"。

相似度描述 i：表明视点与焦点的距离为 di−1 到 di 时物体的相似度描述。它必须描述两方面的信息：

①在此距离区间，覆盖此表面的网格描述。预先计算并存储$\triangle d_i$(i=0，1，…)，以加快处理速度。

②在此距离区间，此网格上每一格的处理描述。

此处，在"整体相似度描述 1"下，物体 A 由 s 个表面构成。不同的相似度描述下，构成物体的面数也不同。一般地，随着 Di 的增加，物体的面数随之减少。这符合整体相似度的描述。

由以上的模型，就能在减少图象生成时间的基础上，对其进行快速操作。

算法描述如下：

While 物体未处理完 do

Begin

根据物体离视点的距离确定整体相似度并进行此相似度下的处理。

While 表面未处理完 do

Begin

计算视点和当前焦点 t 的距离 d。

用 d 在当前表面相似度描述表上进行查找，找到相应的隶属区间后就可得到相应的相似度描述，进而处理：

①得到此表面的网格描述，使得网格的最大面积块包含 t，同时让网格覆盖此表面。

②对网络的每一格，根据相应的每一格相似度描述对它进行处理。

End

End

3 结束语

VR 技术是一门新兴学科，其中许多问题有待进一步解决。本文提供的分析和算法可对系统建模过程进行整体上的控制，使得系统在满足视觉要求的基础上，在物体表示一级上尽可能地提高图象生成速度。

参考文献

1 汪成为.灵境技术是建立人机和谐仿真系统的关键技术.系统仿真学报，1995，7(4)：1−4

2 邵胜利，陈悦，陈瑜申.虚拟现实中的三维建模技术.计算机世界报.专题综述，1995(10):123−125

3 赵沁平，怀进鹏，李波，沈旭昆.虚拟现实研究概况.计算机研究与发展，1996，33(7)：497−500

4 Deering M.High Resolution Virtual Reality.Computer Graphics，1992, 26(2):195−202

5 Krueger MW.Automating Virtual Reality.IEEE Computer Graphics & Applications，1995，15(1)：9−11

(上接第39页)

向一个服务器组插入一个服务(组名，服务名)；
SGMerge 合并多个服务器组(组名表)；
SGRmvGroup 撤消一个服务器组(组名)。

2 结语

Client／Server 是当前软件开发中的重要领域，我们针对 Client 复杂的服务请求，提出了协作计算模型，并在此基础上实现了协作模型开发工具 FD−CDT，正如文中所讲的，我们在 Windows95 环境下用 VisualC++实现了 FD−CDT 的一个原型。为简化服务描述，我们在文中仅给出了基本的服务接口，为进一步实用化，我们正研究将该工具与多种数据库管理系统的结合并进一步完善 Client／Server 开发的系列工具。

参考文献

1 Bal H，Steiner J，Tanenbaum A. Programming Languages for Distributed Computing Systems. ACM Computing Surveys，1989，21(3)：261−322

2 Carriero N，Gelernter D. Coordination Languages and their Significance. Comm. ACM，1992，35(2)：97−107