第16卷 第12期
2006年12月

计算机技术与发展
COMPUTER TECHNOLOGY AND DEVELOPMENT

Vol.16 No.12
Dec. 2006

分布式系统中基于Jini的JavaSpace技术研究

刘 玲,赵 刚,李 丹
(西南石油大学 计算机科学学院,四川 成都 610500)

摘 要:由于要仔细考虑在本地计算中不会出现的很多问题,比如局部实效、等待时延加剧、语言的兼容性等,使得构建分布式系统显得较为困难。Java语言的远程方法调用(RMI)能够解决普遍的分布式计算问题。而JavaSpace技术被设计来解决两大问题:分布式的持久性和分布式算法设计,JavaSpace服务利用RMI及Java语言的串行化特征来实现这一目的。文中对基于Jini的JavaSpace技术进行了详细介绍,深入分析了JavaSpace共享分布式计算模型及其核心技术,介绍了目前JavaSpace技术的各种应用,以在此基础上构建强大的分布式应用系统。

关键词:分布式;Jini;JavaSpace
中图分类号:TP311.5 **文献标识码**:A **文章编号**:1673-629X(2006)12-0061-03

Study of Jini-Based JavaSpace Technology in Distributed System

LIU Ling, ZHAO Gang, LI Dan
(Computer Science Institute, Southwest Petroleum University, Chengdu 610500, China)

Abstract: Distributed systems are hard to build. They require careful thinking about problems that do not occur in local computation. The primary problems are those of partial failure, greatly increased latency, and language compatibility. The Java programming language has a remote method invocation system called RMI that approach general distributed computation. JavaSpace technology is designed to solve two related problems: distributed persistence and the design of distributed algorithms. JavaSpace services use RMI and the serialization feature of the Java programming language to accomplish these goals. Gives a detailed introduction of the technology of Jini-based JavaSpace, and analyses the share-distributed model of computing and the kernel built in JavaSpace. At last the current applications of JavaSpace are given. And can build strong distributed system by these technologies.

Key words: distribution; Jini; JavaSpace

1 Jini和JavaSpace简介

1.1 Jini

Jini是Sun公司的研究与开发项目,它建立在Java平台之上,并能极大地扩展Java技术的能力。Jini技术可使范围广泛的多种硬件和软件(即可与网络相连的任何实体)能够自主联网。Jini的功能是将一个网络上的服务连接并组织在一起,并使客户端能发现和使用这些服务。

1.2 JavaSpace

JavaSpaces是建立在Jini之上的一种技术,它紧密依附于Jini体系结构。JavaSpace作为一种共享分布式通信的机制,还可作为一种存储对象的机制;它提供了永久地建立和保存对象的能力。JavaSpace是一个利用Jini基础结构并向其他Jini客户机和服务提供其功能的一个Jini服务。JavaSpace提供了一种完成共享分布式计算的机制。

收稿日期:2006-03-09
作者简介:刘 玲(1978-),男,四川安岳人,硕士研究生,研究方向为计算机网络、P2P技术、分布式系统;赵 刚,副教授,硕士生导师,研究方向为嵌入式、并行计算、分布式系统。

2 分布式系统中的JavaSpace

在分布式系统中,对象之间应当能够相互通信、共享信息。JavaSpace服务(JavaSpace service)利用对象的分布式存储(distributed repository)和3个简单操作(读、写和取),实现了一个简单的、构建分布式系统的高层体系结构。JavaSpace服务通过Jini事务管理器和通知机制来支持事务,当与某个给定模板相匹配的条目写入JavaSpace服务时,通知机制能够通知某个对象。

2.1 基于Jini的JavaSpace服务

JavaSpace服务为Java对象提供分布式的共享存储器。任何与Java兼容的客户端都可以将共享对象放进这个存储器中。然而,对于这些Java对象有几个限制条件。首先,保存在JavaSpace服务的对象都必须实现接口Entry(net.jini.core.entry.Entry)。

Jini核心规范[1]中定义了Entry对象使用的相关规则。一个Entry对象必须包含一个不需要使用任何参数的构造器方法,这个方法又助于实现对象的序列化;Entry对象只能包含具有public属性的实例变量,在进行相关查询时将使用这些变量;所有实例变量必须是可被序列化的,任何原始数据类型都不能写入JavaSapce或从JavaS-

pace 中读取。

JavaSpace 技术需要几个底层的服务。JavaSpace 服务依赖于 Jini 查询服务,在需要事务时,必须启动 Jini 事务服务,JavaSpace 服务还依赖于一个 Web 服务器和 RMI 活动守护程序 rmid。

2.2 JavaSpace 服务的属性

JavaSpace 技术简化了分布式系统的设计与开发。一个 JavaSpace 服务有 5 个主要的属性:

1)JavaSpace 服务是一种 Jini 服务。

2)一个条目将一直保存在 JavaSpace 服务中,除非它的合约到期,或是由某个程序从 JavaSpace 服务中取出。

3)JavaSpace 服务定位对象的方法是将对象与模板进行比较。模板指定了 JavaSpace 服务比较各个条目的搜索条件。如果有一个或多个条目匹配模板,JavaSpace 服务将返回其中的一个。

4)JavaSpace 服务使用 Jini 事务管理器来支持操作执行的原子性。

5) JavaSpace 服务的对象是共享的。程序可以从 JavaSpace 服务中读出或取得条目,改变这些条目的 public 字段,以及写回到 JavaSpace 服务供其他程序使用。

2.3 永久对象仓库

JavaSpace 的一个特点是对象数据仓库。写入一个空间的项都是正式的 Java 对象。但 JavaSpace 并不是一个对象数据库。Entry 实例在位于 JavaSpace 中时并不是活动的,只是能够作为拷贝访问。这表示不能直接更改空间中的一个项。例如,如果在一个空间的某行中两次写相同的 Entry 实例,则此空间中将会有两个项。因此,空间中不存在维护项的对象标识的概念。

2.4 共享分布式计算模型

租用、事件和事务处理的产生是由分布式程序设计的特性所决定的。相对于一般的本地计算来说,分布式计算环境中的基础环境易出错。JavaSpace 引入了一种不同的模型。

2.4.1 JavaSpace 的松散-耦合模型

JavaSpace 提供了进行通信的一种中介模型(松散-耦合模型[2])。其实现使用了分布式数据结构,使得多个进程以并行方式访问和操作一个对象结构中的内容。图 1 和图 2 显示了这种模型。

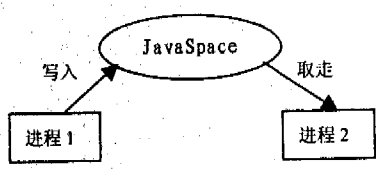

图 1 JavaSpace 的松散-耦合模型 I

该模型的功能是分离了进程。不用操心特定进程通信的细节,进程 1 所要操心的是写一个项到 JavaSpace,进程 2 无需关心项是怎样进入 JavaSpace 的,它只要取走它们,然后做自己的工作即可;对进程进行分离有几个好处,如果进程 2 失败,并不会影响进程 1,进程 1 仍然能够继续完成自己的任务;如果需要添加另一个进程,只需在图中简单地加上它即可(如图 2 所示)。

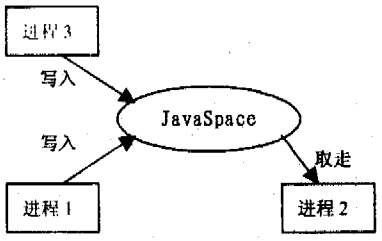

图 2 JavaSpace 的松散-耦合模型 II

图 2 中,进程 3 可以很轻松地向空间写入项了。因为进程 1 不需要知道进程 2 的细节,所以添加进程 3 也不需要更改它,这里进程 2 并不关心空间中的项来自何处,它只需使用它们即可,这就是 JavaSpace 的松散-耦合模型。这种模式在很大程度上降低了分布式程序设计的复杂性。

2.4.2 JavaSpace 空间操作

在 JavaSpace 空间上的基本操作如图 3[3]所示。

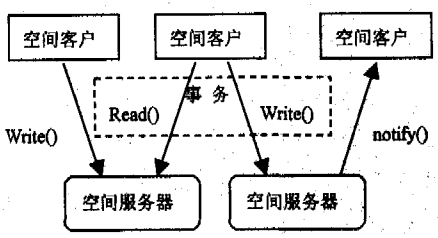

图 3 JavaSpace 上的基本操作

为支持分布式通信,JavaSpace 提供了一种强大却又非常简单的 API。在 net.jini.space 包中为 Jini 的外部实现定义了如图 3 所示的 JavaSpace 接口。就一个基本的操作 Write 语句来说,它把一个 Entry 对象拷贝到一个 JavaSpace 服务中。而将一个入口对象写入到 JavaSpace 空间中的操作也可能会对注册到该 JavaSpace 空间的其他对象生成一个通知消息 notify()。

3 JavaSpace 技术核心

3.1 对象文件系统

JavaSpace 的目标是提供"对象的文件系统",也就是说,JavaSpace 可提供一种无所不在的、自然的方式来存储和使用对象,它的设计是为了自然地工作于用 Java 开发的面向对象应用程序。另外和文件系统相似的是,JavaSpace 可支持在应用之间共享对象,因此就像一个应用(或用户)可以把文件放到文件系统中众所周知的位置以供其他应用(或用户)取出一样,JavaSpace 可作为支持 Java 的客户和服务之间共享的通信中介。但通常的文件系统只存储"字节包",如在 UNIX 和 Windows 系统中,共享文件甚至不与类型相关联。多数情况下,这取决于使用文件的应用及已建立的约定,把某种意义或解释与文件系统中被称为文件的"字节包"联系起来。

而 JavaSpace 是面向对象的存储系统。JavaSpace 不是存储简单的无类型数据,它利用 Java 存储整个对象,以

及作为 Java 对象所具有的好处:强类型、可移动代码、安全执行等。因此在 JavaSpace 中,被存储的实体有实际类型,并且可以包含代码。

JavaSpace 的另一点不同在于寻找被存储实体的方式。在文件系统中,对先前存储数据的访问是通过命名,文件系统中的每个文件都有一个在文件系统中唯一的名称。若这个名称是被使用它的各方周知且认可的,则按照约定,它可以有特殊的属性并且可以被很多其他实体使用。文件系统中的搜寻通常是搜索文件名中的一部分或搜索文件的实际内容(适于部分情况如文本文件)。

在 JavaSpace 中,名称并不重要。事实上,对象的"名称"只是可用来寻找对象的众多属性中的一个。可以基于对象的类、超类,或它们实现的接口来搜寻对象,也可以基于对象的属性搜寻对象。按照约定,任何属性都和名称一样对待。

3.2 基于属性的搜寻

如果对象不必用名字来标识它们,那么客户如何使用 JavaSpace 来存储和找到对象呢? JavaSpace 使用与 Jini 查找服务基于属性的搜寻完全相同的技术,每个存储在 JavaSpace 中的对象都必须实现 net. Jini. core. entry. Entry 接口,意思是它可以被解释为是其内部成员对象的强类型集合。事实上,JavaSpace 和 Jini 查找服务使用属性搜寻的唯一区别,就在于查找服务定义了一组模板如何与一组项目匹配的条件,而 JavaSpace 只有匹配单个项目的工具。我们知道 Entry 是一个无方法的"标记"接口,对象可实现它,告诉系统这个对象的创建者知道此对象被用到查找服务的特殊方式(对象的成员被独立地序列化等)。Entry 对象在 JavaSpace 中也具有完全相同的语义:它们被当作是自己指向的公有、非静态、非最终、非变化对象的集合。每个 Entry 可通过与其成员对象匹配的"模板"被搜寻,搜寻时使用的模板可以基于类型或对象的值,支持"通配符"。

尽管 JavaSpaces 使用不同的 API 集来存储、搜寻和取出对象,但它使用完全相同的基于属性的搜寻规则。它使用的 API 更适用于存储服务的需要。

4 Jini 及 JavaSpace 技术的应用

Jini 提供了在分散式环境中寻找(look - up)、注册(registration)、租借(leasing)等功能。而 JavaSpace 则负责管理分散式物件的处理程序(processing)、分享(sharing),以及流通(migration)等。因此 Jini 与 JavaSpace 彼此存在着相互合作的关系。简单地说,JavaSpace 就好像网路上的一个市场,它提供一个简单、快速、统一的界面,让网路上分散的资源可以被分享、协调与流通。

对于要存储对象或使对象可被其他对象使用的很多应用,JavaSpace 技术可以作为一种新的分布式系统编程范型的基础并提供一种新的用于创建分布式应用的模型。在这种新模型中,分布式系统的创建将使对象在应用之间"流动",通过中心的元组空间进行协同。这种思想不是为新的分布式应用创建定制的远程通信接口或协议,而是应用可根据它们写入元组空间及从元组空间中读出的对象集合来定义。如果用 Java 术语描述它,就是说不是为每个新任务定义或细化新的 RMI 远程接口,而是 JavaSpace 接口将成为应用之间交互的公共 API,应用将定义自己的 Entry 对象集,并给这些对象赋以自己的语义。

JavaSpace 技术的使用领域:

1)信息共享。

JavaSpace 可以共享对象。因此,许多 JavaSpace 的应用程序都可以实现对象的共享。可通过空间方便地交换信息。写到一个空间的每个项都可以从该空间的任一个客户机读出。读取者和写入者不需要互相了解。所需知道的只是何种项放入了此空间。例如,空间可在聊天或通话系统中使用,这种系统已变得非常流行。其中,空间被用作共享的留言板,多个客户机可以写入和读取消息。

2)计算服务。

除共享数据以外,JavaSpace 的一个非常有意义的用途是共享分布式系统计算。一般在这种用法中,计算一个(或多个)问题块的产生器可能很昂贵,但这种问题可以分成并行任务。产生器把问题块发送到某个 JavaSpace,然后,运行在分离的机器上的该空间的客户机使用这些块,进行计算并将完成了的块返回给 JavaSpace。然后再把这些完成的块装配起来。

3)工作流。

JavaSpace 也很适合于管理工作流[4]环境。可以把工作流环境与纯计算服务环境区分开来,在工作流环境中所调度的工作可以由人而不是纯计算完成。

4) P2P[2]软件代理中使用 JavaSpace 实现数据的分布式存储。

5 总　结

JavaSpace 是一个 Jini 服务,它提供一种共享分布式对象仓库。JavaSpace 是用 Java 所发展的技术,并且以 RMI 实现其网路通讯的功能,一般应用在 n - tiers 架构的中间层(middle tiers)。JavaSpace 虽然能提供供求者与供应者之间查询与沟通的机制,但它并不是资料库,而是以简单的 messaging system 为基础,进而提供更强大的功能。除了在 Jini 应用之外,JavaSpace 技术也可被应用在其它系统与服务中,如: Workflow systems, Customer management systems, Supply chain management, Intelligent rich data distribution, Trading services, Auction systems, Resource allocation and management systems, Agent Systems, 以及 Publish and subscribe services 等[5]。因此,进一步进行 JavaSpace 的具体应用研究是非常有意义的。

参考文献:

[1] Edwards W K. Jini 核心技术[M]. 北京:机械工业出版社,

(下转第 66 页)

表 3 扩展后的场景权重因子表

事务的数目	权重因子
1	5
2	5
3	7.5
4	10
5	10
6	12.5
7 或多于 7 个	15

3 分解执行者估算单个用例

通过分解执行者权重的方法实现对单个用例的估算，设 $UUCP_i$ 表示第 i 个用例的未调整用例点，$UUCW_i$ 表示用例的未调整权重，$RUAW_i$ 表示与第 i 个用例相关的执行者权重。设有 n 个执行者是该用例的主执行者，其中每一个执行者可能是 m 个用例的主执行者，则与第 i 个用例相应的未调整执行者权重($RUAW_i$) 可以计算得到：

$$RUAW_i = \sum_{j=1}^{n} UAW_j * \frac{UUCW_i}{\sum_{k=1}^{m} UUCW_k}$$

其中 $\sum_{k=1}^{m} UUCW_k$ 是 UAW_j 所代表的执行者作为主执行者的所有用例场景的权重和，则第 i 个用例的未调整用例点 $UUCP_i = RUAW_i + UUCW_i$。

4 案例研究

4.1 估算结果

结合几个项目的实际数据，对几种估算方法进行了对比，表 4 是这几个项目的估算和结果比较。

表 4 项目估算表和估算结果比较

项目	编号	实际值(人时)	用例点(人时)/错误率	专家经验(人时)/错误率	扩展用例点(人时)/错误率
统计局业务系统	SHSIS	10994	7380/32.87%	8056/26.72%	7421/32.5%
Lucky ERP System	LuckyERP	10336	8642/16.39%	8816/14.7%	8428/18.46%
办公自动化系统	MHOA	6688	6240/6.7%	8080/20.81%	6459/3.4%
房产交易系统	HTS	2972	4428/49%	3800/27.86%	4158/39.91%

4.2 估算结果分析

* 对比专家经验估算和用例点估算。

4 个项目中有两个项目专家经验的错误率比用例点方法低 5 个百分点以上，有一个专家经验比用例点方法高 10 个百分点以上，有一个项目两种方法大致相等。总体而言，用例点估算方法是一种有效的估算方法，可以和专家经验结合使用，当两种估算方法结果偏差较大时可以考虑重新估算。

* 对比扩展用例点估算和用例点估算。

扩展用例点方法和用例点方法估算的结果比较接近，这是由于扩展用例点方法是源自用例点方法的。在 4 个项目中有 3 个项目扩展方法的错误率比用例点方法低，而且从 4 个项目综合来看，扩展用例点方法也比用例点方法具有略高的准确度。所以，在使用用例点方法的场合，可以使用扩展用例点方法的用例复杂度权重因子表(见表 3)来替代用例点方法中的权重因子表(见表 1)，以获得更高的准确度。

* 估算单个用例。

在单个用例的估算上，对比专家经验和扩展用例点方法可以发现两者的差异与两者对项目整体估算的差异是大致相等的。所以，扩展用例点方法对单个用例估算和对项目整体进行估算具有基本一样的准确度。

5 结 论

文中的主要研究成果在于在研究估算技术、用例、用例点估算方法和模糊集的基础上提出了两个用例点估算方法的扩展：一是使用用模糊集扩展用例点方法，扩展后的方法将比用例点方法具有更好的准确度，并且不会使原用例点方法变得复杂；二是通过把角色的权重分解到各个用例，从而实现对单个用例的估算。经过与专家经验以及实际值的比较，发现对单个用例的估算的准确度和对整个项目估算的准确度是大致相等的，这说明扩展用例点方法对单个用例的估算是有效的、可靠的。

参考文献：

[1] Kishore S. 软件需求与估算[M]. 北京：机械工业出版社，2004.

[2] Jacobson. Object - Oriented Software Engineering: A Use Case Driven Approach[M]. New York: Addison - Wesley Publishing Company, 1992.

[3] Karner G. Metrics for Objectory[D]. Sweden: University of Linköping, 1993.

[4] Ribu K. Estimating Object - Oriented Software Projects with Use Cases[D]. Norway: University of Oslo, 2001.

[5] Damodaran M, Washington A N E. 用例点估算方法[J]. 非程序员，2005(45)：61 - 65.

[6] Zadeh L A. Fuzzy Sets[J]. Information and Control, 1965, 8: 338 - 353.

[7] de Souza Lima Júnior O, Farias P P M, Belchior A D. A Fuzzy Model for Function Point Analysis to Development and Enhancement Project Assessments[J]. CLEI Electronic Journal, 1990, 5(2): 1 - 14.

(上接第 63 页)

2002.

[2] Flenner R, Abbott M. Java P2P 技术内幕[M]. 北京：人民邮电出版社，2003.

[3] Boger M. Java 与分布式系统[M]. 北京：机械工业出版社，2003.

[4] 朱文华，王 茜. 基于 Jini 技术和 JavaSapce 构建分布式工作流管理系统[J]. 计算机应用研究，2003.(6)：82 - 84.

[5] Newmarch J. A Programmer's Guide to Jini Technology[M]. [s.l.]: Apress, 2000.