虚拟企业 CORBA 分布式组件 调度 信息集成

计算机科学2000Vol. 27№. 5

26-29,34

虚拟企业环境下基于 CORBA 的分布式组件调度*)

Study on CORBA Based Scheduling for Distributed Components in Virtual Enterprises

杨晓春 张利荣 于 戈 王国仁 郑怀远

(东北大学计算机研究所 沈阳110006)

TP391.7

Abstract Scheduling for distributed components is an advanced issue in virtual enterprises research areas, nowadays. This paper reviews the developing of integration and compares the five different component scheduling architectures based on CORBA in details. Finally it presents three component customization methods for non-definite scheduling architecture.

Keywords Virtual enterprise, Scheduling, CORBA, Component object, Component customization

1 引言

信息集成技术一直是企业为增强竞争而追寻的一种新的信息管理技术,而组件调度的思想也正来源于此。但伴随着 Internet 和虚拟企业的发展,传统的信息集成技术已不再能满足新的需求。

传统集成方法一般归纳为两类:点对点转换方法[1]和公共数据模型转换方法[2]。

◇点对点转换方法如图1(a)所示。系统要使用一些精确的转换规则或算法,将一个模型表示的模式转换成用另一个模型表示的模式。这种方法的缺点是转换算法是特定的,不能用于所有源和目标模型的转换,而且在任意两个模型间的转换都需要两个算法,例如从模型 A 到模型 B 的转换和从模型 B 到模型 A 的转换。

◇公共数据模型转换方法如图1(b)所示。这种方法使用公共的中介模型代替原来从源模型到目标模型的直接转换,从而避免开发大量的不可共享的转换方法。它将转换方法分为两步:源模型首先转换成中介模型,然后再转换成目标模型。

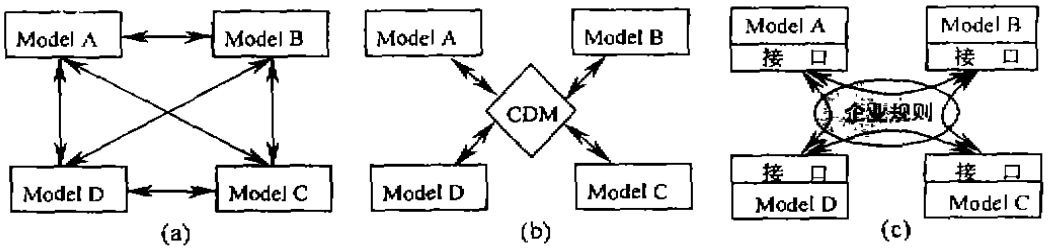

图1 解决模型异构性的几种集成策略

在虚拟企业中,人、信息、材料等通过 Internet 被关联在一起,随着应用变得越来越复杂,越来越分布,以上两种方法均不适用于大规模资源共享的需要,不但转换代价高,而且随着分布式环境的发展,这种转换也是不实际的,不可能要求每个对象都先转换成符合公共数据模型(CDM)的信息,同时也很难找到一个合适的 CDM 用于支持各种异构模型的转换。因此,需要采用一种新的方法来解决分布式对象间的信息共享问题。

◇组件定制方法,如图1(c)所示。这种方法要求采用不同数据模型的各组件(成员系统)都要按照统一的接口协议来定义自己的接口,形成分布式环境中的组件对象。接口支持对组件对象的访问,而组件对象内部采用的模型对接口外部的应用是不可见的。这样组件

*)本课题得到863/CIMS 主题(863-511-946-003)和第六届霍英东青年基金资助。

对象间只通过彼此提供的接口来互相访问。

与前两种方法相比，第三种方法具有如下优点：(1)低复杂性。对于N个数据模型，点对点转换方法需要N*(N-1)个特定的转换算法，公共数据模型转换方法需要2*N个转换算法，而组件定制方法只需定义N个接口。(2)高扩展性。如果向一个已拥有N个模型的系统中加入一个新的模型，只需要加入1个新的接口定义，而用点对点转换方法或公共数据模型转换方法都需要增加2*N个新的转换算法。

组件定制方法的引入为虚拟企业的信息共享提供了有效的策略，CORBA的问世为这种思想带来了勃勃的生机。CORBA提供了对象总线ORB、接口定义语言IDL和各种服务，使其为构造分布式应用提供了一个清晰的、高级的途径。它比套接字(Sockets)和PRC更加成熟，通过ORB解决平台异构性，可以很好地满足虚拟企业对于健壮性的需求。

我们从使用IONA公司的Orbix和DEC公司的OB中，获得了一些有关CORBA产品的使用经验，通过对分布式环境下对象管理技术及相关工作流管理系统结构的考察，我们归纳出五种支持分布式组件集成的调度结构，并逐一进行分析和比较。另外，本文还讨论基于CORBA的三种组件定制方法。

2 组件对象的调度结构

在虚拟企业这种分布式动态系统中，组件对象以一种松耦合的形式彼此联系、相互制约。其中，与对象调度密切相关的组件包括调度器、字典、监控器、日志记录和组件对象本身。显然，组件对象是一个企业的动态实例，它们通过CORBA IDL被定义为CORBA对象，可以独立运行，也可以交织在一起共同完成某个工作。调度器根据这些组件对象间的依赖关系协同地调度各组件对象以便满足某个企业逻辑，字典负责向调度器提供与调度相关的模式信息，在实际系统中，各系统保存模式信息的介质不同，有使用文件的，也有使用数据库的，在此我们将这些存储介质统称为字典。监控器主要负责监控组件对象的访问状态。为了便于全局监督和恢复，提供日志记录负责记载整个调度的执行情况。在调度过程中，根据调度器对整个调度流程的掌握程度，虚拟企业环境下基于CORBA的动态调度结构被归纳为确定型和非确定型两种，又因调度器与字典所在的物理位置，分为集中式、半分布式和分布式几种情况，下面分别进行讨论。

2.1 确定型集中式调度

在这种结构中[3]，调度器进程支持多线程服务。调度器被设置在某个中心节点上，是一个主线程，产生许多子线程与组件对象对应。全局字典与调度器处在同一节点，为调度器提供与调度有关的模式信息，在各节点上的组件对象作为被调度的单元，提供CORBA IDL接口与调度器通讯，其结构如图2所示。

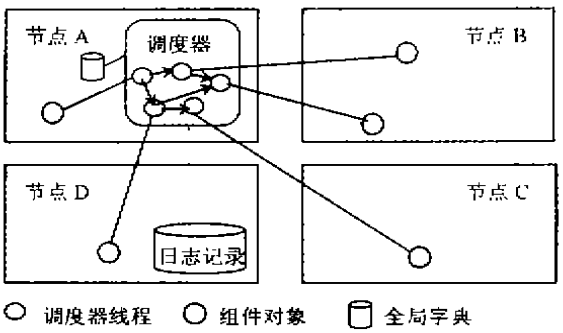

图2 确定型集中式调度

确定型集中式调度分为同步调度和异步调度两种方式，当以同步方式调度时，调度线程发请求给对应的组件对象后，调度线程阻塞，等待执行结果的返回；而采用异步方式调度，在被调度的组件对象执行过程中，调度器线程不阻塞。CORBA提供两种异步方式：oneway和延迟同步。使用oneway时，只调用组件对象的方法，而不接收任何返回值、out或inout参数。当执行结束后，组件对象再通过回调函数将结果通知给调度对象。延迟同步支持动态调用接口(DII)的异步操作，对于被调用的服务器是透明的。从调度器的角度看，使用延迟同步更方便些，但要求组件对象要采用DII来定义。不同的系统可以根据具体情况来选择不同的方式。

集中式结构实现简单，易于维护，但在大型的应用领域内可能存在潜在的调度瓶颈问题，影响系统的性能。

2.2 确定型分布式调度

确定型分布式调度不限制调度对象的定位，如图3所示，没有集中调度器，每个调度对象作为独立进程可以分散在不同节点上，彼此通讯，在每个调度器所处的节点上都有一个局部字典负责记录与本地组件对象有关的调度模式，一个核心日志记录用于收集和监控各对象的执行情况。调度对象是由IDL定义的CORBA对象，通过IDL接口与被调度的组件对象一一对应。在这种调度结构中，所有的调度对象组成了整个系统的调度结构，每个调度对象根据它前面的调度对象传递的信息决定是否、何时调度自己所管辖的组件对象的执行。通常，调度对象采用同步IDL接口与被管理的组件对象通讯。文[4]中的系统采用了这种调度结构。

确定型分布式调度要求每个调度对象在初始化时必须明确自己的职责，知道如何调度它的后续对象。适用于企业逻辑的内在分布特性，同时消除了集中式调度的通讯瓶颈问题。

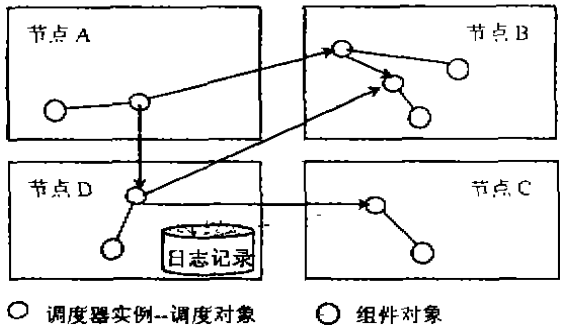

图3 确定型分布式调度

确定型分布式调度有一种特例，将调度代码直接加入到每个组件对象中，使各组件对象成为主动对象，知道在接收到什么样的消息（事件）时启动自己，下一步该触发哪个对象，并在执行结束时及时通知给日志记录，这种方式从调度角度来看是最有效的方法。由于涉及到的对象数目少，使整个系统的负担大大减轻。但与确定型分布式调度一样，每个对象必须要事先知道自己的调度模式。而且，一旦企业逻辑发生变化，每个组件对象都要进行重新包装，以满足新的企业逻辑的需求。这种方式适用于规模较小的、调度模式长期不变的虚拟企业。

以上两种确定型的调度方式更适用于虚拟企业的工作流管理，在一个工作流中定义了整个调度过程的确定的模式信息（包括工作流的开始、结束标志、各种跃迁条件等），这些模式信息被保存在字典中，为调度器提供调度依据。因此，调度器可以在宏观上对整个流程的调度过程进行控制。

2.3 非确定型集中式调度

从调度器的角度来看，调度器对于整个调度过程没有一个清晰的概念，在全局字典中只保存一些系统预先定义好的规则，记录组件对象被调用时应触发的下一个动作是什么，即应调度哪个对象执行后继动作。其中，在每个节点的每个服务器上提供一个监控器，用于监控该服务器上所有组件对象的执行状态（包括哪个操作被调用，何时被调用，返回的结果是什么）。监控器将被监控对象的执行状态通知给调度器，调度器匹配全局字典中的规则，若匹配成功，触发相应的规则，调度组件对象的下一个动作。实现时，借助 CORBA 提供的 filter 来构造监控器。

非确定型调度中，调度器是典型的异步工作方式，它只起到一个中转站的作用，用于匹配并触发规则，非确定型调度实施动态调度，更适用于企业逻辑变化频繁的情况[5]。

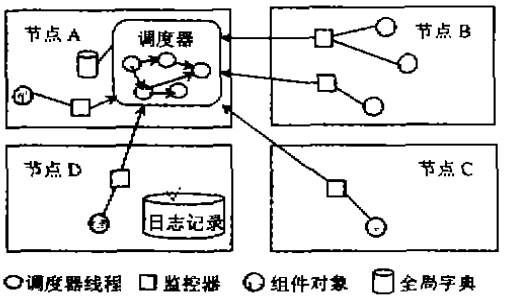

图4 非确定型集中式调度

2.4 非确定型半分布式调度

非确定型半分布式调度将调度对象分散在不同的节点上，但提供一个全局唯一的字典来记录组件对象间的各种联系，这些联系以规则的形式存在。其中，与全局字典在同一个节点上的调度对象被称为主调度对象，它不仅调度节点 A 上的组件对象，还维护一个调度对象管理表，记录其它调度对象的分布情况。主调度对象通常采用异步通讯方式，在完成调度任务的同时，当有新的企业规则被定义时，还要检查这些新写入的规则，对照调度对象管理表，将新的模式信息及时发布给各调度对象，以指导其它调度对象的调度。在文[6]中采用了非确定型半分布式调度结构，如图5所示。

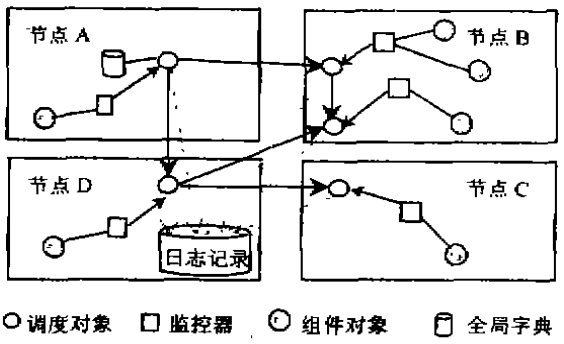

图5 非确定型半分布式调度

非确定型半分布式调度将调度对象进行分布，解决了集中式结构中调度器作为瓶颈的缺陷，但是，集中访问的全局字典仍然是系统性能的一个障碍。另外，这种结构加重了主调度对象的负担，往往会导致严重的负载不平衡。

2.5 非确定型分布式调度

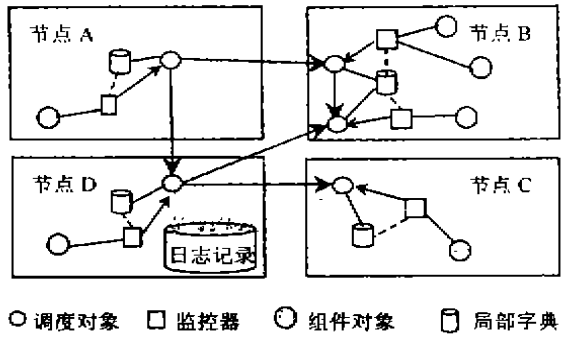

图6 非确定型分布式调度

图6显示了非确定型分布式调度的示意图，与非确定型半分布式调度的区别在于，将全局字典中的调度信息分散在各节点上，保存在局部字典中，局部字典中的信息并不是对全局字典的简单拷贝，而是只保留与本地调度相关的模式信息[7]。

为了更好地提高性能，监控器可以访问本场地的局部字典，只有被过滤到的组件对象的动作与局部字典中记录的事件匹配时，才通知调度对象进行调度。采用这种方式，减轻了调度对象的负担，使各对象（包括调度对象、监控对象、组件对象）间和各节点间的负载平衡得到较好的改善。

总结上述四种动态调度方式，比较列表如下：

结构 / 项目	确定型 集中式调度	确定型 分布式调度	非确定型 集中式调度	非确定型 半分布式调度	非确定型 分布式调度
适用范围	局部网（小型虚拟企业），工作流管理	广域网（大规模虚拟企业），工作流管理	局部网（小型虚拟企业），动态企业调度	局域网（中小型虚拟企业），动态企业调度	广域网（大规模虚拟企业），动态企业调度
可伸缩性	差	差	好	好	好
字典维护	简单	一致性维护较困难	简单	简单	要考虑字典复制问题
负载平衡问题	差	较好	差	较差	较好
实现程度	容易	较难	容易	较难	难
CORBA 通讯方式	同步或异步	同步	异步	主调度对象异步、其它调度对象同步	同步或异步

3 非确定型调度的组件定制方法

虚拟企业中的大部分组件对象（成员系统）是被事先定义好的，用于支持某个企业的特定功能的一些遗产应用。非确定型调度结构中，为支持组件对象的灵活调度，使其在新构筑的虚拟企业中很好地协同工作，要对这些组件进行重新定制，以满足虚拟企业中新的企业逻辑的需求。基于 CORBA 的组件定制方法可以概括为以下三种：

·代码重写。在最坏的情况下，我们简单地将原代码重写去定制组件的行为，以满足虚拟企业的需求。使用代码重写的前提条件是假设我们能够获得组件中类和方法的实现代码。当然，这是组件定制方法中最不希望用到的方法。

·代码映射。如果组件的开发者有远见，可以将组件的功能设计成适于组件彼此可以调度的。于是，组件可以被包装成新的组件对象，为组件对象定义接口，实现组件方法名到接口的映射，如图7所示。这种方法通过组件对象的特征值或操作来间接地定制组件。代码映射包括特征映射和行为映射两种方式。

①特征映射。用组件对象的特征值来驱动组件的行为。主要用于支持基于模式的集成，用 IDL 为每个组件定义对象包装接口，并在接口处提供一组特征（如公共属性）去映射组件提供的方法名。这些在接口定义中的特征可以被直接访问。于是在集成阶段，组件对象中的属性可被设置成某个值来通知其它组件工作。JavaBeans 就采用了这种方法。

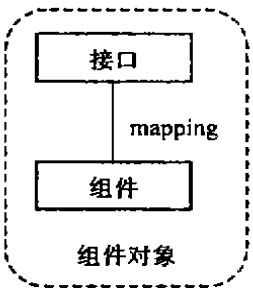

图7 代码映射

②行为映射。用组件对象的操作来驱动组件的行

（下转第34页）

```
unregister_agent();
unregister_agent_system();
unregister_place();
};
```

参 考 文 献

1 Lange D B, Oshima M. Seven good reasons for mobile agents. CACM, 1999, 42(3): 88～89

2 Wong D, et al Java-based Mobile Agents. Some to[1], 92～102

3 Shoham Y. Agent-oriented programming Artificial Intelligence, 1993, 60: 51～91

4 Nardi B A, Miller J R Collaborative, Programmable Intelligent Agents. CACM, 1998, 41(3): 96～104

5 Shehory O, Kraus S. Methods for task allocation via agent coaliton formation. Artificial Intelligence, 1998, 101: 165～200

6 OMG. CORBAservices. Common Object Services Specification. Nov. 1997

7 OMG. The Common Object Request Broker: Architecture and Specification for CORBA V2. 2. Feb. 1998

8 Vinoski S. New features for CORBA 3. 0. CACM, 1998, 41(10): 44～52

9 Crystaliz, Inc. General Magic, Inc IBM Corporation, et al. Joint Submission. Mobile Agent Facility Specification. June 1997

10 刘锦德. 对于开放系统内涵的澄清. 计算机应用, 1997, 17(6): 1～3

11 苏 森, 唐雪飞 开放系统中的互操作性. 计算机应用, 1997, 17(6): 4～7

12 苏 森. 面向对象的互操作技术. [电子科技大学博士论文]. 1998

13 陈 羽中, 麦中凡. 可移动的软件 Agent 研究. 计算机科学, 1998, 25(5): 62～66

14 陶先平, 吕建, 等. 流动 agent: 一种未来的分布计算模式. 计算机科学, 1999, 26(2): 1～4

(上接第29页)

为。用 IDL 为每个组件定义对象包装接口，并在接口处提供一组操作去映射组件提供的方法名。在接口处提供的操作通常是简单的数据访问，当获知组件对象中的某些数据发生变化时，调度其他组件对象进行工作。

·规则定制。这是最灵活的一种方法。组件中的每个方法都代表了组件的功能行为。我们可以引入主动数据库中的 ECA 规则：在一个组件中发生的事件被其他组件所关注，当一个事件发生时，该组件发送(Post)另一个事件来通知其他组件。被发送的事件触发一组规则，当规则的条件满足时执行相应的一组动作。事件与事件触发规则可以被用于定制代码，从而有效地定制组件的行为。此时，事件是钩链(hook)，而规则是必要的代码。基本上，事件和事件触发规则提供了一种能够添加和替换代码的机制，在某个方法的实现中加入或替换部分代码而不必访问源代码。因此，规则定制是最合适的一种方法。

结束语 综上所述，虚拟企业环境下分布式对象的集成问题的核心是如何调度各组件系统协同工作以满足不同企业的企业逻辑。本文在回顾了集成系统发展过程的基础上，总结了分布式组件的几种动态调度结构，对其进行了全面的比较，并提出了在非确定型调度结构中的组件定制方法。

参 考 文 献

1 Ozkarahan E. Database management concepts, design and practice, Chapter 10. Prentice Hall, 1990

2 Yang X, et al. Study on a CORBA-based Plug and Play Mechanism for Distributed Information Sharing Environments: [SCOPE Technical Report]. Dept. CS, Northeastern University, Sept. 1998

3 Noll J, Scacchi W. Supporting Software Development in Virtual Enterprises. Journal of Digital Information, Dec. 1998, 1, issue 4

4 Berger M. CoNus- A CSCW System Supporting Synchronous, Asynchronous and Auronomous Collaboration. In: Proc. Intl. Conf. on Distributed Platforms/Industrial Stream, Dresden, Feb 27-March 1, 1996

5 Lan H, Su S Y W. Component Interoperability in a Virtual Enterprise Using Events/Triggers/Rules.

6 Morrie M C, et al. CIM through Database Coordination. In: Proc. of the Int. Conf. on Data and Knowledge Systems for Manufacturing and Engineering, Hongkong, May 1994

7 Yang X, et al. VIASCOPE: an CORBA/IIOP-based Information Integration System for Virtual Enterprises. In: Proc. of the Int. Conf. on ICYCS' 99. Aug. 1999