关于 COM + 组件边界和同步的研究与分析*

和　力[1,3]，吴丽贤[2]，鱼　滨[1]，和　燕[3]

(1.西北大学 计算机科学系，陕西 西安 710069；2.韩山师范学院 数学系，广东 潮州 521041；3.楚雄师范学院 计算机科学系，云南 楚雄 675000)

摘　要：引入了围绕在 COM + 对象周围的三种边界环境:进程、套间和上下文,分析了实现跨越这些边界调用的机制,同时论述了 COM + 在不同环境中实现并行保护的策略。

关键词：组件；COM + ；套间；上下文；同步域

中图法分类号：TP311　　**文献标识码**：A　　**文章编号**：1001-3695(2002)10-0017-02

Study and Analysis on the Boundaries and Synchronization of COM + Component

HE Li[1,3], WU Li-xian[2], YU Bin[1], HE Yan[3]

(1. *Dept. of Computer Science, Northwest University, Xi'an Shanxi 710069, China*; 2. *Dept. of Mathematics, Hanshan Teachers' College, Chaozhou Guangdong 521041, China*; 3. *Dept. of Computer Science, Chuxiong Normal University, Chuxiong Yunnan 675000, China*)

Abstract: In this paper, process, apartment, context-the three execution contexts around the COM + component are presented. It analyzes the mechanism to realize the calls across the boundaries of these execution contexts, and meantime, discusses the strategies of realizing synchronization protection under various environments.

Key words: Component; COM + ; Apartment; Context; Synchronization Domain

1　引言

可以将 COM + 看作是 COM,DCOM 和 MTS 的集成和发展,由此把 COM 组件提升到应用层而不再是底层的软件结构,将底层细节留给操作系统,使得 COM + 与操作系统的结合更加紧密。COM + 的底层结构仍然是以 COM 为基础,但在应用方式上更多地继承了 MTS 的处理机制,包括事务机制、安全模型、对象池等;同时新增了一些服务,比如负载平衡、事件模型、队列组件等,使 COM + 成为适合构建多层分布式的组件技术。

COM + 程序设计方法的核心是基于对属性值进行声明请求的模型。COM + 运行时系统将根据组件设置的属性来实例化对象,并把对象装入相匹配的执行环境里运行。我们可把执行环境看作是一组具有相容属性的对象运行时所处的环境,它相当于给运行其中的对象构筑一道边界。每一个 COM + 对象在外围具有三道这样的边界:进程、套间和上下文。在它的整个生存期,它只能生存于某一进程、某一套间和某一上下文中,也就是说一个 COM + 对象不能越过自己的边界进入另一执行环境中。当进行跨边界(包括跨进程、同一进程中跨套间和同一套间中跨上下文)调用时,调用者和被调用的对象之间必须加入代理机制,只有处于同一上下文中的对象才能进行直接调用。虽然多线程技术可以显著地提高应用程序的性能,但隶属于同一进程的线程由于可以对共享地址空间内的数据进行并行访问,这将很容易产生竞争情况。竞争情况可以通过对共享数据实施串行化访问,即进行同步处理来加以避免。在 COM + 中有两种方法可实现对对象的串行化访问:①由系统自动实现对 COM + 对象成员函数的串行化访问,它保证下一次方法调用在上次调用完成后才可进行;②使用事件、互斥变量、监界区和信号量等机制来保护共享数据。针对处于不同边界中的对象和各种跨边界调用,COM + 采用不同的策略来进行同步处理。

2　跨进程调用与同步

由于不同的进程使用不同的地址空间,所以客户不能直接调用进程外组件(包括本地进程外组件和远程组件)。对于跨进程调用,中间必须经过代理/存根(Proxy/Stub)机制和使用列集/散集(Marshaling/Unmarshaling)手段。如图 1 所示,代理对象在客户进程中扮演组件对象,它具有与组件对象一样的接口,但不提供方法实现。客户实际调用的是代理对象,代理对象负责把调用参数及其它一些调用信息组装成一个数据包,并通过 RPC/QC 通道列集给组件进程中的存根代码;存根代码负责把包散集到组件进程的地址空间中,并进行实际的接口功能调用,函数的返回值和输出参数按相反的顺序传回

收稿日期：2001-12-25；**修返日期**：2002-04-11

基金项目：国家自然科学基金资助项目(60073050)；陕西省教委专项基金资助项目(99JK142)

给客户。

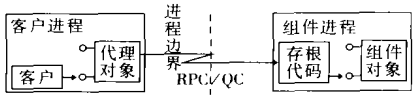

图 1 跨进程调用

对于进程外组件对象，客户调用必须跨越进程边界，因此调用始终是间接进行的，中间必须经过调度，而这种调度自动实现了对调用的串行化访问，对象成员函数可以不处理同步。

3 套间与同步

为了适应多线程的需要，COM + 为此提供了一种称为套间的结构。我们可以把套间想象成一个盒子，里面装着线程和组件对象，每一个要使用 COM 的线程和每一个这些线程创建的组件对象都必须属于某一个套间。一个套间只能属于某一个进程，且不能跨越进程边界。

目前 COM + 支持三种套间，它们是单线程套间（Single-Threaded Apartment，STA）、多线程套间（MultiThreaded Apartment，MTA）和中性套间（Thread-Neutral Apartment，TNA）。一个进程可以包含一个 MTA、一个 TNA 和多个 STA。

一个 STA 中只能容纳一个线程，但 COM 对于里面包含的组件对象个数并未限制，STA 中的对象只能由该套间中惟一的线程来调用。当线程调用 CoInitialize( ) 或 CoInitializeEx(NULL, COINIT _ APARTMENTHREADED) 函数时，COM 将构建一个新的 STA，并把这个线程赋予新创建的 STA。同时自动地调用 CreateWindowEx 来为 STA 创建一个隐藏窗口，客户对套间中对象的调用被当作一个发给此隐藏窗口的消息，套间线程主函数的消息循环负责接收消息并分发消息，于是隐藏窗口会接收到消息，然后它的窗口过程调用对象的相应接口成员函数。通过这种消息分发机制自动实现了对 STA 中对象调用的同步，所以运行在 STA 中的组件对象不必处理同步，但对象仍需保护全局变量，因为一个组件可以在多个套间中创建实例，它们有可能被同时访问。

MTA 中的线程没有个数的限制，也没限制所包含的对象个数，任何线程都可以通过调用 CoInitializeEx (NULL, COINIT _ MULTITHREADED) 函数来进入 MTA。MTA 没有隐藏窗口也没有消息队列，MTA 中的组件对象可供任意线程访问，COM 并不保证对组件对象的串行化访问。这样带来的问题是，同一个组件对象可能同时被多个客户线程调用，因此组件本身必须通过使用互斥体、事件、信号量和监界区等手段对共享数据进行保护，也就是说 MTA 的组件对象必须是线程安全的。

TNA 是 COM + 中引进的一种新套间类型，TNA 只作为对象的容器，不包含线程。STA 和 MTA 中的线程调用 TNA 中的对象时，客户线程将暂时离开 STA 或 MTA，然后在 TNA 上执行调用，TNA 中对象的成员函数将在客户线程上直接执行，因此调用 TNA 中的对象不会引起线程切换。与 MTA 情形类似，TNA 中的对象可被任意线程直接调用，因此 TNA 中的组件对象也必须是线程安全的。

值得注意的一点是：对于进程内组件在多线程环境下，不论它使用何种套间类型，其入口函数如 DllGetClassObject 和 DllCanUnloadNow 可能被同时访问到，因此必须进行同步，尤其对引用计数包括对象引用计数和锁计数器需要进行同步保护，类厂根据使用策略也需要进行同步保护。

对于从 STA 到 STA 和 STA 与 MTA 之间的调用，由于调用者和被调用对象分属不同的线程，虽然这两个线程使用同一个进程地址空间，但它们使用不同的堆栈，所以调用如同跨进程调用一样，中间也要通过代理/存根模块和列集/散集的手段间接进行。由于 STA 和 MTA 中的线程可以进入 TNA 中直接调用其中的对象，因此调用不会涉及线程转换。但对于这种情况 TNA 中的对象与调用者位于不同的套间，必然分属不同的上下文环境，因此调用不可避免地要跨越上下文边界，所以中间仍需经过轻量级代理进行环境切换。它们之间的关系可用图 2 表示。

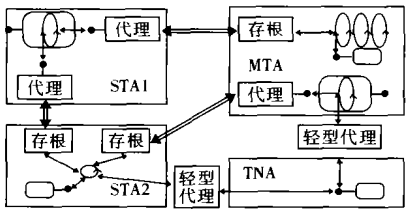

图 2 跨套间调用

上面的讨论中并未涉及 TNA 对象调用其它对象的情况，对于这种调用仍需遵守套间和上下文规则。例如运行在某 STA 线程上的 TNA 对象调用位于另一个 STA 或 MTA 中的对象时，中间仍需经过代理/存根模块并进行线程转换。

4 上下文与同步域

COM + 用来跟踪记录有一个对象的运行属性的数据被引用为该对象的上下文。上下文代表了一个对象生存的周边环境。每个被激活的对象都与一个上下文相关联。上下文可以被看作是对象的最小生存边界，每个上下文只能驻留在一个套间中，而每一个套间可包括多个上下文，一个上下文可以与多个对象相关联。当激活一个对象时，若它与调用者的属性和所需服务相容时，它将在调用者的上下文中被激活；否则将在另一上下文中被激活。

当对象在上下文中处于激活状态时，COM + 对于来自上下文内的方法调用的处理和对于来自于上下文之外的方法调用的处理是不同的。在相同的上下文中由一对象调用另一对象不需要 COM + 的介入，调用将直接进行，并且不涉及线程转换和环境切换。在同一套间中，不同上下文中的对象之间的运行环境是不兼容的，因此跨越上下文边界的调用需要 COM + 采用一种称为监听器（Interceptor）的轻量级代理来进行环境切换，并消除这种不兼容性。图 3 说明了这种情况，对象 A 调用同一上下文中的对象 B 是直接进行的，不需 COM + 监听介入，而对象 B 调用另一上下文中的对象 C 却（下转第 22 页）

综合的、可伸缩的电子商务应用系统的网络计算结构，它具有许多优点：易维护、扩展性强、运营效率和安全性高、可重用性强、资源利用率高、开发效率高。

5 发展的趋势及将来的应用前景

随着信息技术和社会经济的发展，人们都迫切希望商务信息化，企业家和网络经济倡导者们更希望传统的商务大规模地转向信息时代的电子商务，实现从传统的交易费获利向增值服务的转变。人们预测，将来电子商务市场将通过增加布置更丰富的功能来吸引成员而获利，这些丰富的功能集成了多种交易机制和广泛范围的增值服务，因而走出传统通过扩大规模获利的思维。这使得电子商务市场面临新的挑战，包括增加竞争力、所提供服务的快速商业化、开发新的市场、电子商务技术上的复杂性、增加网络效应、更加健壮性。因此，新一代的电子商务所提供的解决方案，应能够支持动态、灵活伸缩性能、协作处理等过程实时管理和跟踪整个商务流程链。电子商务系统应使人们能够在任何时间、任何地点通过任何设备例如电话、传真和电子邮件访问到数据和应用；而且电子商务系统要具有更大的灵活性，即商务模型必须能够随着新的商业需求和机遇易于变换。

基于 M/V/C 的电子商务给商家和企业迎接商务市场信息化提供了一个比较好的解决方案。随着商务市场的合并及新的商机的出现，基于 M/V/C 的电子商务很容易适应这一变化，并支持新的增值服务的能力，因为这种结构具有很好的扩展性和适应性。由于它是开放的、基于标准的体系结构，它能够帮助商业企业和商业集团在整个商业运作过程中便于所有的参与者能够集成协作处理，加速商业信息流动、周转。

参考文献：

[1] http://www.redbooks.ibm.com/ Websphere redbooks.pdf [EB/OL].2001-09-25.

[2] http://www.w3.org/TR/xp-reqs/XML Protocol(XMLP)requirements.html[EB/OL].2001-03-25.

[3] http://www.ibm.com/e-business/infrastructure.html and emarketplace.pdf[EB/OL].2001-09-27.

[4] http://www.oasis-open.org/specs/ OASIS/ebXML Registry Information Model (RIM) v2.0.html[EB/OL].2001-12-03.

[5] www.object-arts.com/EducationCentre/Overviews/MVC.htm (Model-View-Controller framework)[EB/OL].2001-12-03.

[6] Goldfaub, Peud Prescod. The XML Handbook[M]. America: Arentice Hall,1998.

[7] John Arerley.Developing and E-business Application for the IBM WebSphese Application Server[M]. International Technical Support Organization,1999.

[8] G Cornell,C S Horstmann.Cone Java[M].Sunsoft Hall,1996.

作者简介：

许文韬，工程师，硕士，研究方向为网络应用与电子商务；顾君忠，教授，博士生导师，研究方向为 CSCW 与分布式数据库技术。

(上接第 18 页)却需 COM + 通过监听器介入。

同步域是 COM + 引入的一种不依赖套间的基于属性的同步机制。一个同步域可以包含多个上下文，并可跨越和穿过多个套间和进程。在任何时候只能有一个线程执行同步域中某个对象的方法函数，这样就保证了同步域中的对象被串行化访问，并实现同步。任何组件可在 COM + 浏览器中通过声明同步域属性进入父对象所在的同步域或者一个新的同步域中。图 4 给出了进程、套间、上下文和同步域之间的关系。对象 A，B，C，E 和 F 将受到同步保护，而不论它们所属的套间类型，对象 D 不受同步域的并行保护，但如果它驻留的套间类型为 STA，它依然会受到 STA 提供的同步保护。

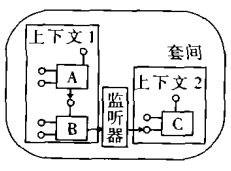

图 3 上下文边界

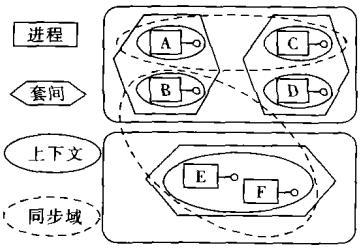

图 4 进程、套间、上下文和同步域之间的关系

运行于 STA 中的组件对象不需要进行同步处理，但对它只能进行串行化访问；而运行于 MTA 中的组件对象可以并行访问，但组件自身必须保证是线程安全的，我们很难在追求效能和简化程序设计这两者之间保持平衡。COM + 引入的 TNA 和同步域可以很好地解决这个问题，我们可以声明组件的线程模型为中性套间，这样的组件对象既可以通过编码保证自身的线程安全性，也可以通过声明同步域属性由 COM + 自动提供同步保护；另一方面 TNA 中的对象可被同一进程中的任何线程直接调用，不需经过额外的线程转换和列集。

5 结束语

微软的组件技术从 COM 发展到 COM + ，为了适应多线程环境和 COM + 所倡导的基于属性的编程模式，提出了上下文、套间和同步域等概念，大大简化了组件开发和程序设计的复杂性。但套间和上下文在进程里面又给对象加上了两条边界，使对象之间的相互调用可能需要跨越更多的边界，需要 COM + 更多地介入，影响调用效率；所以我们必须小心地配置和部署组件，在追求效率和简化程序设计方面尽量保持平衡。

参考文献：

[1] Guy Eddon, Henry Eddon. Inside COM + Base Services[M]. 北京：北京希望电子出版社，2000.

[2] Robert J Oberg. Understanding & Programming COM + [M]. 北京：电子工业出版社，2001.

[3] 潘爱民. COM 原理与应用[M]. 北京：清华大学出版社，1999.

[4] Ash Rofail, Yasser Shohound. Mastering COM and COM + [M]. 北京：电子工业出版社，2000.

作者简介：

和力(1971-)，男，讲师，在职硕士研究生，主要研究方向为组件技术、分布式应用；吴丽贤(1974-)，女，助教，在职硕士研究生，主要研究方向为组件技术、可视化研究；鱼滨(1964-)，男，副教授，硕士生导师，主要研究方向为构件技术、分布式计算、多媒体技术；和燕(1968-)，女，讲师，主要研究方向为计算数学。