CORBA 构件模型综述*

潘慧芳，周兴社，於志文
(西北工业大学 计算机学院，陕西 西安 710072)

摘 要：随着计算机网络技术和应用的发展，分布构件技术成为分布式计算领域的热点，CCM 就是主流的分布构件技术之一。首先介绍了 CCM 产生的背景，然后对 CCM 的重要组成部分进行了详细的阐述，并对现有的基于 CCM 的研究和实现进行了简要的分析，最后将 CCM 与 EJB 和 COM 进行了比较。

关键词：构件；CORBA；CCM
中图法分类号：TP311 **文献标识码**：A **文章编号**：1001-3695(2005)05-0014-02

An Overview of CORBA Component Model

PAN Hui-fang, ZHOU Xing-she, YU Zhi-wen
(School of Computer Science, Northwestern Polytechnical University, Xi'an Shanxi 710072, China)

Abstract: Along with the development of computer networks and applications, distributed component technology became the hotspot of distributed computing. CCM is one of the popular distributed component technologies. This paper first introduces the background of CCM. It then describes the main sections of CCM and analyzes the existing implementations based on CCM. Finally, it compares CCM with related technologies.

Key words: Component; CORBA; CCM

随着网络技术的飞速发展，单个计算节点的处理能力持续提高，不同厂商和异构技术环境的不断激增，使得分布式系统的应用和开发日趋复杂。在构建企业分布式应用系统的过程中，要求系统具有可配置性、可伸缩性、可重用性和可管理性，以满足不断增长的企业应用需求。在这种情况下，分布构件技术应运而生。通过采用分布构件技术，可以降低大型分布式系统的开发难度，重用已有的代码资源，提高分布式系统的开发效率。目前，主流的服务器端分布构件技术有 OMG 组织的 CCM (CORBA Component Model, CCM) 技术，Microsoft 的 COM (Component Object Model, COM) 技术以及 SUN 的 EJB(Enterprise Java Bean, EJB) 技术。因为 CORBA 采用远程对象调用机制，支持异构环境下分布式应用系统的开发和互操作，具有与底层硬件、操作系统、网络、通信协议和编程语言无关的特点，所以被广泛地应用于大型的分布式系统中。而 CCM 作为 CORBA 3.0 规范的一部分，对原有的对象模型进行了扩展，从而更易于服务器端软件的重复使用和 CORBA 应用程序的动态配置，因而具有广阔的应用前景。

1 CORBA 构件模型

传统的 CORBA 对象模型(CORBA2.x 规范) 具有一些明显的缺陷，如没有配置对象实现的标准方式，缺少对公共 CORBA 服务器编程模式的有效支持，对象功能难以扩展，CORBA 对象服务的可用性没有预先定义，对象生命周期管理没有标准化等。这些缺陷导致对象实现难以设计、重用、配置、管理和扩展。为此，OMG 在 CORBA 3.0 中引入了 CCM，CCM 是用于开发和配置服务器端分布式应用的构件模型。

下面对 CCM 中的抽象构件模型、构件实现框架(CIF)、容器编程模型、打包与部署模型进行详细的描述。

1.1 抽象构件模型

CCM 构件提供了称为端口(Ports) 的多种外部接口，以便与客户、其他构件、CORBA 服务等进行交互。构件模型支持四种基本的 Ports。

(1) 侧面(Facets)。它是构件提供的与客户交互的相互独立的一组接口。一个构件能够提供多个对象引用，这些不同的对象引用被称为 Facets，Facets 可以支持不同名字和功能的 IDL 接口。客户通过唯一的等价接口(Equivalent Interface) 在构件的多个 Facets 间进行导航。Facet 接口的实现被封装在构件中，被看作是构件的一部分。

(2) 插口(Receptacles)。它是一些指定的连接点(Connection Points)，这些连接点描述一个构件使用外部构件提供的对象引用来调用其上的操作的能力。通过使用插口，构件能够与其他对象进行连接，并调用这些对象的操作。

(3) 事件源/事件接收器(Event Sources/Event Sinks)。它是指定发送/接收特定类型事件的连接点。事件源分为 Emitter 和 Publisher 两类，Emitter 规定在某个时间只允许一个接收者与之连接，Publisher 允许同时有多个接收者与之连接。事件接收器允许有多个事件源与之相连。

(4) 属性(Attributes)。属性主要用于构件的配置，配置工具使用属性对构件的配置参数进行预先的设置。

CCM 引入了产地(Home) 对构件的生命周期进行管理。一个 Home 是某种类型所有构件实例的管理器，不同类型的 Home 能够管理同一类型的构件，但一个构件实例只能有一个 Home 实例。Home 形式化了工厂(Factory) 设计模式来管理同

收稿日期：2004-05-19；修返日期：2004-06-26
基金项目：国防预研基金资助项目(413160102)

一类型所有构件实例的生命周期。Home 接口提供创建(Create) 和删除(Remove) 构件实例的操作, 客户能够访问 Home 接口来控制他所使用的构件实例的生命周期。客户要使用构件, 必须首先通过 Resolve_initial_references 获得 HomeFinder 接口的对象引用, 再使用 HomeFinder 获得 Home 接口的对象引用, 最后通过调用正确的操作来创建或找到所需构件的对象引用。

1.2　构件实现框架(Component Implementation Framework, CIF)

构件实现框架为构造构件实现定义了编程模型。CIF 引入了构件实现定义语言(Component Implementation Definition Language, CIDL) 来描述构件以及构件 Home 的实现。支持构件的 ORB 产品编译 CIDL 描述文件, 生成实现构架(Skeleton) , 它自动完成了构件的许多基本行为, 如导航、身份查询、激活、状态管理、生命周期管理等。开发者对构架进行扩展, 添加一些用户定制的构件行为, 以创建完整的构件实现。作为一种编程的抽象, CIF 被设计成与现有的 POA 框架兼容, 并且为编程人员屏蔽了使用复杂性。特别是 CIF 能够使用现有的 POA 框架实现, 但又没有直接暴露 POA 框架的任何元素。

1.3　容器编程模型(Container Programming Model)

容器编程模型是一个 API 框架, 它定义了一组 API 接口以简化开发和配置 CORBA 应用的复杂度。容器编程模型的组成如下:

(1) 外部 API(External API) 。定义了构件客户可用的接口, 这些接口被存在接口库(Interface Repository) 中, 由构件开发者实现。

(2) 容器 API(Container API) 。定义了构件开发者使用的 API 框架, 它由内部(Internal) 接口和回调(Callback) 接口组成。内部接口由容器提供, 用来帮助实现构件的行为, 构件可以通过内部接口访问容器支持的服务; 回调接口由构件实现, 被容器调用, 用来在容器内配置和管理构件。

(3) CORBA 使用模型(CORBA Usage Model) 。定义了容器和 POA, ORB, CORBA 服务等的交互。CORBA 使用模型由策略(Policy) 控制, 策略规定了与 POA 和 CORBA 服务的交互模式。

(4) 构件的种类(Component Category) 。有四种 CORBA 构件, 分别是服务(Service) 构件、会话(Session) 构件、过程(Process) 构件和实体(Entity) 构件。

容器编程模型的体系结构如图 1 所示。

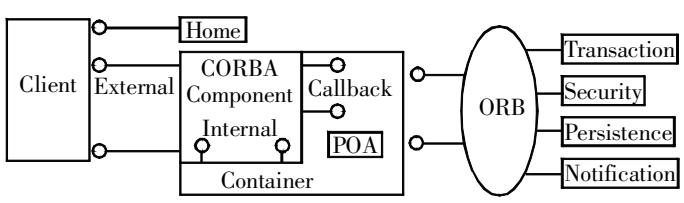

图 1　容器编程模型的体系结构图

容器为 CORBA 构件提供运行时的执行环境。主要功能有: 负责运行时期所有构件实例的创建和管理; 管理 POA 策略以决定如何创建构件引用; 为常用的 CORBA 对象服务(交易服务、安全服务、持久性服务和通知服务) 提供了一个适配层, 从而将客户的请求转发给对应的 CORBA 服务。

1.4　打包和部署

CCM 规范使用 XML 来指定构件的打包和部署信息, 以简化构件在大型分布式系统上的配置和使用。

构件包(Component Package) 是用于部署的最小单元, 它包括构件的描述信息和一组二进制的构件实现文件。描述信息在运行时供容器使用; 每个实现文件对应在一种硬件平台、操作系统、语言编译器和 ORB 上编译产生的二进制代码。

一个构件包可以被单独部署, 也可以被包含到一个组装包(Assembly Package) 内和其中的其他构件一起部署。组装包由一个包描述文件、一组构件包和属性文件组成。包描述文件说明该组装包由哪些构件组成以及它们之间的相互关系。

使用构件包或组装包部署构件的过程如下: ①部署工具与用户交互确定构件安装到哪些目标主机上; ②将构件实现安装到目标主机上; ③在主机上实例化构件 Home 和构件; ④对于组装包, 还需按照组装包描述信息建立构件之间的连接关系。

2　CCM 研究与实现系统

目前, 基于 CCM 的研究和实现有很多, 其中比较有代表性的有: 法国里尔大学 GOAL 项目组开发的 OpenCCM(Open CORBA Component Model Platform) , 美国华盛顿大学的 CIAO (Component-Integrated ACE ORB) , CPI 开发的 EJCCM(Enterprise Java CORBA Component Model) 以及 MICO 项目组开发的 MicoCCM。

OpenCCM 是第一个免费的、开放源码的 CCM 实现。OpenCCM 是完全使用 Java 语言开发的, 所以支持跨平台的移植。OpenCCM 可以运行在 JacORB, ORBacus, OpenORB 等多个 CORBA 产品上, 支持 Linux, Solaris, Windows 等操作系统。OpenCCM 由开放产品工具链、打包和组装工具链、分布式部署设施、容器运行框架和管理框架组成。因此, OpenCCM 允许用户设计、实现、编译、部署和执行分布式 CCM 应用程序。

CIAO 是建立在 TAO 之上的 CCM 实现。CIAO 致力于为分布式实时嵌入式系统(DRE) 的开发人员提供一个面向构件的范例。CIAO 的研究集中在将 QoS 需求、实时策略等 DRE 的关键问题抽象成在构件组装中便于描述的元数据(Metadata) , 从而将 QoS 需求等非功能部分与构件实现相分离, 增加 DRE 系统的灵活性和重用性, 简化系统的配置和管理。

EJCCM 是对 OMG 的 CCM 的 Java 实现, 并且也是开放源码的。它具有如下特性: 定义了构件(包括属性 Attributes 和端口 Ports) 和构件 Homes, 支持基本的持久性功能, 构件容器支持服务构件、会话构件、过程构件和实体构件四种类型, 提供分布式的部署功能(能够部署远程主机的构件包 JAR 文件) , 能够自动地创建、配置和连接分布式的构件。

MicoCCM 项目始于 2000 年 10 月, 它致力于在开放源码的 ORB 产品 Mico 的基础上提供一个 CCM 的参考实现。MicoCCM 具有以下特性: Mico IDL 编译器被扩充为支持 IDL 3; 支持所有类型的 Ports; 支持多种服务以及会话容器; 提供了安装、加载、管理构件实现的工具; 组装和部署工具包支持构件组装包(包含多个构件和配置信息的组装包可以通过一步操作完成部署) 以及 XML 描述符。

3　CCM 与 EJB, COM 的比较

EJB(Enterprise Java Beans) 是开发和部署面向对象分布式企业级应用程序的另一主流的构件模型。EJB 技术允许开发者不必从头编写所有的代码, 只需链接预先生成(下转第 26 页)