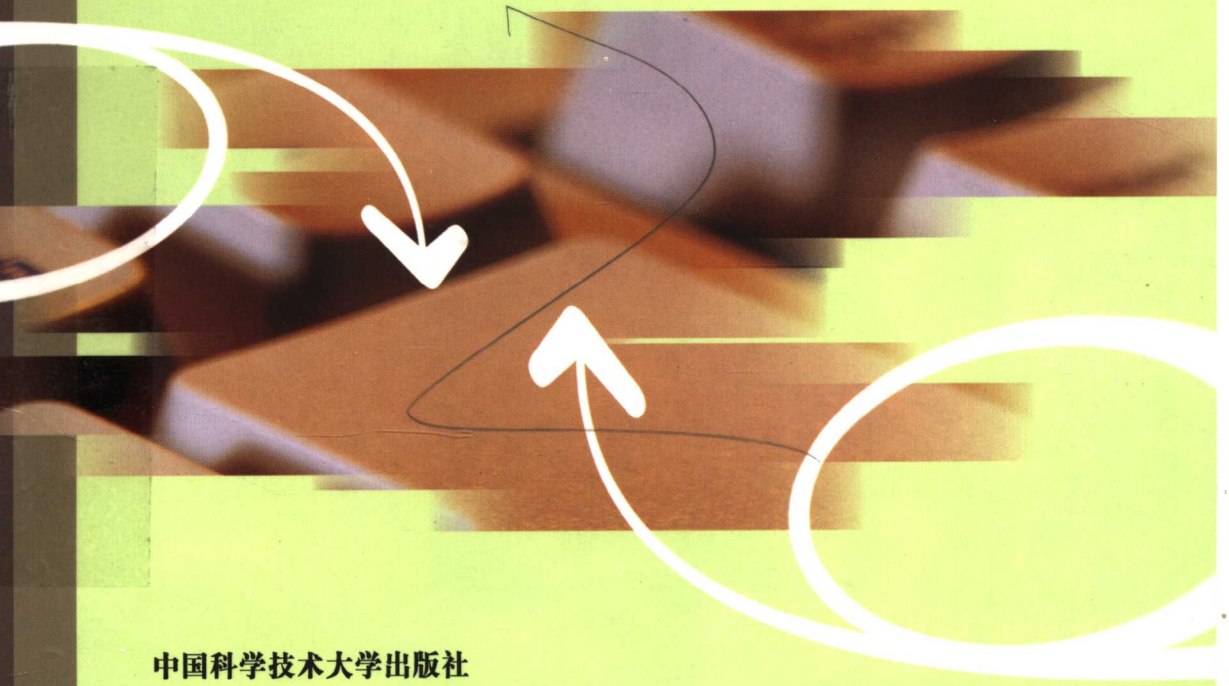
异构组件互操作技术研究

博士论丛

张驰 著

Research on Heterogonous Components Interoperability Technique



中国科学技术大学出版社

张驰 著

异构组件互操作技术研究

Research on
Heterogonous Components
Interoperability Technique

中国科学技术大学出版社

图书在版编目(CIP)数据

异构组件互操作技术研究/张驰著. —合肥:中国科学技术大学出版社,2008.3

ISBN 978-7-312-02191-6

Ⅰ.异… Ⅱ.张… Ⅲ.软件开发—研究 Ⅳ.TP31

中国版本图书馆 CIP 数据核字(2007)第 192979 号

出版 中国科学技术大学出版社
安徽省合肥市金寨路 96 号,230026
网址:http://press.ustc.edu.cn
印刷 安徽江淮印务有限责任公司
发行 中国科学技术大学出版社
经销 全国新华书店
开本 710 mm×1000 mm 1/16
印张 9.5
插页 1
字数 152 千
版次 2008 年 3 月第 1 版
印次 2008 年 3 月第 1 次印刷
定价 18.00 元

前　言

在1968年北大西洋公约组织的软件工程会议上，Mcllroy第一次提出了软件复用的概念。1983年，Freeman对软件复用给出了详细的定义——“在构造新的软件系统的过程中，对已存在的软件人工制品的使用技术”，此后随着对计算机软件研究的不断深入，面向对象技术不断发展，软件复用受到人们越来越多的关注。

软件复用是指重复使用为了复用目的而设计的软件的过程，而可复用软件则是指为了复用目的而设计的软件。软件复用出发点是应用系统的开发不再采用“一切从零开始”的模式，而是以已有的工作为基础，充分利用过去应用系统开发中积累的知识和经验，从而将开发的重点集中于应用的特有构成成分。

复用是成熟的工程领域的一个基本特征，传统产业发展的基本模式都是标准的零部件(组件)生产与基于标准组件的产品生产相结合。其中组件是核心和基础，复用是必需的手段。实践证明，这种模式是产业工程化、工业化的必由之路。

基于组件的软件开发方法旨在复用预先开发的、经过验证的软件组件构造应用系统，大大减少软件开发中人力、物力的投入，缩短开发周期，提高软件的质量。这是实现软件复用的切实可行的途径，也是软件走向工程化和产业化的必由之路。

由于现代软件系统规模越来越大，复杂度越来越高，随着对软件复用研究的不断深入，基于组件的软件开发思想逐渐深入人心。为了简化系统开发复杂度，人们将事务逻辑和系统逻辑分离，从而使事务开发人员能致力于事务逻辑的开发，而系统逻辑由软件供应商提供。基于组件的软件开发技术以软件架构为组装蓝图，以可复用软件组件为组装模块，支持组装式软件的复用，大大提高了软件生产效率和软件质量。

人们将处理事务逻辑和处理系统底层的基础设施分离，形成了不同的分布

式中间件标准，在中间件技术之上提出了组件模型，用来说明如何构造组件以及组件间如何进行交互。如今主流的组件模型有 OMG 组织的 CORBA 组件模型、Microsoft 的 DCOM 组件模型和 SUN 公司的 EJB 组件模型，这些组件模型构成了企业应用的基础。由于没有统一的组件标准，所以目前存在着多种组件模型，这些异质的组件有其各自的特点，这包括不同的应用平台、不同的实现语言、不同的技术特点等，在 Internet 环境中，在一个集成商业程序的所有参与者间确保一个单一、统一的体系架构是十分困难的，组件技术的目标本来是为了能够更好地实现代码重用，但缺少异构组件间的互操作，用户就不得不为不同的分布式组件系统编写相同功能的代码。对于企业应用来说，基于不同规范的异构分布式组件的互操作成为一个非常重要的特性。不同组件模型的组件如何进行互操作对提高软件复用程度，实现企业内部应用集成和跨企业交互具有十分重要的意义。

本书对异构组件互操作相关问题进行了深入研究，全书共分为 6 章。第 1 章为绪论，分析了目前主流分布式组件模型的技术特点和实现异构组件互操作的主要方法。第 2 章为组件模型与异构组件互操作研究现状，指出了在开放环境下组件接口定义语言在组件的描述和互操作方面的不足。第 3 章为组件接口扩展和组件描述。提出了组件接口规约框架和基于 XML 的可扩展的组件标准化描述模板，根据提出的组件接口规约框架对组件接口进行了扩展。基于契约化设计思想对组件接口进行了语义层信息扩展，基于多元 π 演算对组件接口进行了行为协议层信息扩展，从而使组件接口全面准确地描述了组件的“静态”和“动态”信息。第 4 章为基于扩展接口的组件匹配和交易服务，在扩展接口的基础上研究了组件匹配和组件交易服务。在接口的语法层提出了基于类型的基调匹配，在语义层提出了基于前件/后件和一阶谓词逻辑的语义层匹配，在协议层提示了基于 π 演算互模拟型理论的协议层匹配。在研究了组件匹配后，提出了基于标准化描述模板的组件交易服务，研究了基于协作服务器图模型 CSG 的交易者联邦动态管理问题。第 5 章为组件组装与推导，研究了组件组合和组合互操作中的组装推导，给出了在体系结构指导下从候选组件集合中对组件的配置算法，研究了互操作中的关键问题——兼容性和替代性。在兼容性方面依据 π 演算中兼容性理论和角色模型，研究了角色划分、角色正确性、角

色兼容性等，在替代性方面，依据 π 演算中行为继承性的关系理论，提出了行为子类的概念，并结合一个电子商务实例进行了说明。第 6 章为基于 Web 服务的异构组件互操作，Web Services 的出现为异构组件互操作提供了新的契机，设计了以 Web Services 框架为基础的异构组件互操作实现方案。并以 CORBA 组件为例设计并实现了集成网关，讨论了实现集成的关键技术——接口映射、消息类型转换和 CORBA 对象的动态调用。

本书由张驰独立撰写，在成稿过程中得到了西北工业大学计算机学院吴健教授、胡正国教授、蒋立源教授的辛勤指导。在出版过程中得到了江西财经大学科研处和软件学院的大力支持，也得到软件学院各位领导和老师的大力支持，尤其是夏家莉院长、钟元生教授、尹爱华副院长、刘细发主任、李华旸副院长、黄茂军博士等，衷心感谢他们的关心和帮助。最后，衷心感谢妻子聂萍女士和亲人们多年来的鼓励和支持。

由于作者学术水平有限，疏漏和错误在所难免，恳请读者批评指正。

张　驰

2007 年 10 月

目　录

第1章 绪 论

1.1 研究背景

1.1.1 软件复用

在北大西洋公约组织的软件工程会议上，Mcllroy第一次提出了软件复用的概念. 1983年，Freeman对软件复用给出了详细的定义——“在构造新的软件系统的过程中，对已存在的软件人工制品的使用技术”[1,2]. 此后，随着对计算机软件研究的不断深入，面向对象技术不断发展，软件复用受到人们越来越多的关注.

软件复用是指重复使用为了复用目的而设计的软件的过程，而可复用软件则是指为了复用目的而设计的软件. 软件复用的出发点是，应用系统的开发不再采用“一切从零开始”的模式，而是以已有的工作为基础，充分利用在过去应用系统开发中积累的知识和经验，从而将开发的重点集中于应用的特有构成成分[2].

首先，软件复用能够提高软件生产率，减少开发代价. 其次，用可复用的组件构造系统还可以提高系统的性能和可靠性. 因为可复用组件大都进行过高度的优化，并在实践中经受过检验，通过复用这些高质量的已有成果，能避免开发中可能引入的错误和不当，可以控制软件开发的复杂度，缩短开发周期，从而提高系统的质量. 第三，软件复用能够减少系统的维护代价. 第四，软件复用能够提高系统间的互操作性，由于系统实现的不一致性，要实现组件的复用，系统应当有效地解决与其他系统之间的互操作性问题. 第五，软件复用能够支持快速原型设计. 第六，软件复用还能减少培训开销.

复用是成熟工程领域的一个基本特征. 传统产业发展的基本模式大都是标

准的零部件(组件)生产与基于标准组件的产品生产相结合.其中组件是核心和基础,复用是必需的手段.实践证明,这种模式是产业工程化、工业化的必由之路[1,8].标准零部件业的独立存在和发展是产业形成规模经济的前提.计算机硬件产业的成功发展就是依据这种模式,实践已充分证明这种模式的可行性和正确性,因而是软件产业发展的良好借鉴.软件产业要发展并形成规模经济,标准的生产和组件的复用是关键因素.依据复用的方式,可以将软件复用分为"黑盒"复用和"白盒"复用,"黑盒"复用指对已有组件不需任何修改直接进行复用,是理想的复用方式[2].

依据复用的对象,可以将软件复用分为产品复用和过程复用.产品复用是指复用已有的软件组件,通过集成组装得到新系统.过程复用是指复用已有的软件开发过程,使用可复用的应用生成器来自动或半自动生成所需系统.过程复用依赖于软件自动化技术的发展,目前只适用于一些特殊的领域.目前最现实、最主要的软件复用方式是产品复用,即复用已有的软件组件.近几十年来,面向对象技术出现,逐渐成为主流技术,为软件复用提供了基本的技术支持.探讨应用系统的本质,可以发现其中通常包含三类成分:通用基本组件、领域共性组件和应用专用组件.应用系统开发中的重复劳动主要集中于前两类构成成分的重复开发[1].

软件复用已经融入软件工程研究的主流,被视为使软件开发真正走向工程化和产业化道路的希望.总的来说,软件复用有三个基本问题:一是必须有可复用的产品,二是所复用的对象必须是有用的,三是复用者要知道如何去使用被复用的对象.软件复用还包括两个相关过程,即可复用软件产品的开发和基于可复用软件的应用系统构造[2].解决好这几个方面的问题才能实现真正意义上的软件复用.基于组件的复用是目前学术界和产业界公认的主流复用技术,与其他复用方式相比,基于组件的复用更可行、更实用.

1.1.2 基于组件的软件开发

近年来软件开发技术的一个重要进展就是组件化.由于现在的软件系统规模越来越大,要求一个系统完成的功能越来越多,因此软件复用和系统集成具有非同寻常的意义.基于组件的软件开发技术是新一代软件技术发展的标志.

基于组件的开发(component-based software development,CBSD)或基于组件的软件工程(component-based software engineering,CBSE)是一种软件开发新范型,它是在一定组件模型的支持下,复用组件库中的一个或多个软件组件,通过组合手段高效率、高质量地构造应用软件系统的过程[3-10]. CBSD 的理论建立在软件工程、软件复用和分布式计算基础之上,重点研究构件的互操作性(interoperability)、可用性(usability)、可复用性(reusability)和效率(efficiency)等.

由于以分布式对象为基础的组件实现技术日趋成熟,CBSD 已经成为现今软件复用实践的研究热点,被认为是最具潜力的软件工程发展方向之一[4]. 软件不仅在规模上快速发展,而且其复杂性也在急剧增加. 使用软件复用技术可以减少软件开发活动中大量的重复性工作,这样就能提高软件生产率,降低开发成本,缩短开发周期. 同时,由于软件组件大都经过严格的质量认证,并在实际运行环境中得到检验,因此,复用软件组件有助于改善软件质量. 此外,大量使用软件组件,软件的灵活性和标准化程度也可望得到提高.

CBSD 遵循"购买而不创建(buy, don't build)"的开发哲学[3],使人们从"一切从头开始"的程序编制方式转向软件组装,将以往"算法+数据结构"的开发方法转化为"组件开发+组件组合"模式. 基于组件的开发任务包括创建、检索和评价、适配、组装、测试和验证、配置和部署、维护和演进以及遗产系统再工程等主要活动[12]. 图 1-1 是 CBSD 的主要活动,它们与传统的软件开发生命周期中的方法不尽相同.

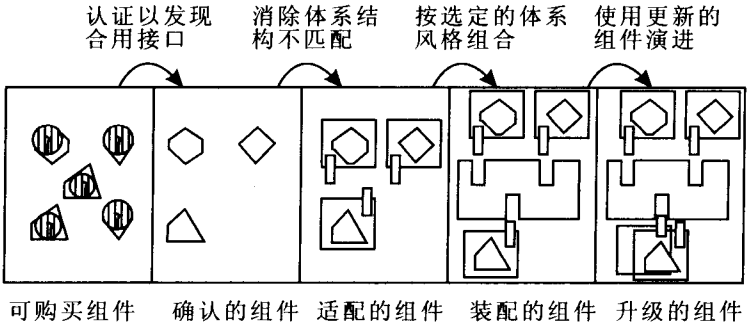


图 1-1 基于组件的软件开发过程[12]

对象是人们提出重用性方面的第一个尝试，组件技术是在面向对象软件开发方法的基础上逐渐发展起来的一种新技术. 由于面向对象开发环境本质上不够完善，缺乏解决对象互操作的公共基础设施，妨碍了它们成为主流产品. 组件技术与面向对象技术紧密相关，组件和对象都是对现实世界的抽象描述，通过接口封装了可复用的代码. 不同的是：首先在概念层上，对象描述客观世界实体，组件提供客观世界服务[13]；其次在复用策略上，对象通过继承实现复用，而组件通过合成实现复用；最后在技术手段上，组件通过对象技术而实现，一个组件通常是多个对象的集合体.

可见，组件技术和面向对象技术之间的关系如下：① CBSD 抽象了许多面向对象技术的实现概念，是将面向对象技术应用于系统设计级上的一种自然延伸. ② CBSD 是面向对象的一个简化的版本，它注重封装性，但忽略了继承性和多态性. ③ CBSD 是构造系统的体系结构级的方法，并且可以采用面向对象方法很方便地实现组件. 有时一个组件就是一个对象，但一般意义上组件好比是化合物，而对象好比是原子，组件能被直接使用，而不必关心构成它们的原子. ④ 在面向对象框架下实现软件重用一般须提供源代码，而用组件实现则完全不用了解它是如何实现的.

CBSD 的兴起主要源于以下 4 个背景：① 在研究方面是对现代软件工程思想，特别是对复用技术的强调；② 在产业方面是支持用组件来建造 GUI、数据库和应用部件的一些理论上质朴但实际可用的技术的成功；③ 在策略方面是某些主流互操作技术，如 CORBA、COM 和 EJB 的开发者的推动；④ 在软件界，对象技术的广泛使用，提供了建造和使用组件的概念基础和实用工具[14-15].

组件是指语义完整、语法正确和具有可重用价值的单位软件，是软件重用过程中可以明确辨识的系统；结构上，它是语义描述、通信接口和实现代码的复合体. 简单地说，组件是具有一定功能，能够独立工作或能够同其他组件装配起来协调工作的程序体，组件的使用同它的开发、生产无关[16]. 从抽象程度来看，面向对象技术已达到了类级重用(代码重用)，它是以类为封装的单位. 这样的重用粒度还太小，不足以解决异构互操作和效率更高的重用. 组件将抽象的程度提到一个更高的层次，它是对一组类的组合进行封装，并代表完成一个或多个功能的特定服务. 整个组件隐藏了具体的实现，只用接口提供服务.

软件组件是指应用系统中可以明确辨识的有机构成成分，是软件系统设计中能够重复使用的构造模块，是一个独立的、可替换的系统的一部分，它具有相对独立性、互换性和功能性[4]. 软件组件不依存于某一个系统，它可以被相同功能的组件所替换，并且具有实际的功能意义. 其特征有：① 有用性，必须提供有用的功能；② 可用性，必须易于理解和使用；③ 可靠性，组件自身及其变形必须能正确工作；④ 适应性，应易于通过参数化等方式在不同的语境中进行配置；⑤可移植性，能在不同的硬件平台和软件环境中工作.

软件组件技术是一种社会化的软件开发方法，它使得开发者可将用不同语言、由不同供应商开发的组件组装在一起来构造软件系统. 通过软件组件的社会化生产，可实现软件的社会化生产，在全球范围内实现软件生产的协作分工. 生产软件组件的公司以各种高质量的标准将可组装的软件组件投放市场，而应用软件开发商们在全球范围内采购和订购可重用的软件组件，再通过组装和加工，生产用户所需的应用软件.

总之，可以预见，组件化软件生产将会对整个软件界产生巨大的影响，并会推动软件产业进一步发展. 而组件技术所扮演的角色就是把零件、生产线和装配运行的概念运用在软件工业中，利用它可以提高开发速度，降低开发成本，增加应用软件的灵活性，降低软件维护的费用. 软件组件技术是软件产业化革命的必然发展趋势[4].

1.1.3 软件体系结构的出现和发展

1996年第1届软件体系结构国际研讨会(International Workshop on Software Architecture, IWSA-1)召开，标志着软件体系结构作为软件工程的一个研究分支正式提出. 作为控制软件复杂性、提高软件系统质量、支持软件开发和复用的重要手段之一，软件体系结构自提出以来，日益受到软件研究者和实践者的关注. 软件体系结构的定义是：软件系统的基本组织，包含组件、组件之间、组件与环境之间的关系，以及相关的设计与演化原则等[184]. 软件体系结构与基于组件的软件开发过程密切相连，它提供了一种自顶向下、基于组件的软件系统开发途径，将软件系统分解为一组组件及组件之间交互的关系，从而使得软件开发者可以从全局的角度去设计和分析系统，克服了传统的基于组件的软件

开发过程中采用自底向上途径的局限.

在软件体系结构的指导下,选择合适的可复用组件进行组装,可以在较高层次上实现系统,并能够提高系统实现的效率.在构件组装过程中,软件体系结构设计模型起到了系统蓝图的作用.现阶段通过组件组装实现软件体系结构设计模型主要关注两方面的内容:① 如何支持可复用组件的互联,即对软件体系结构设计模型中规约的连接子的实现提供支持;② 在组装过程中,如何检测并消除体系结构失配问题.

从应用的角度来看,目前较为成熟的组件技术通过接口描述语言来规范组件之间的交互,但接口描述语言一般仅能定义交互接口中符号级方面的内容,即接口中的参数个数、顺序和类型等方面的内容,而对于组件之间正确交互必需的行为协议方面的规定则无法加以保证.因此,对于功能接口完全匹配的两个交互组件而言,在运行时仍会因行为协议的不相容而引发失败.因此,文献[185]认为,组件间能够正确组装交互不仅需要定义符号级的规范,还需要定义协议级的交互规范,即组件间接口操作调用次序的约束与规范.所以在设计基于组件的软件体系结构过程中,需要考虑通过某种有效途径来描述和验证软件体系结构中组装组件正确交互所必需的行为相容性及软件体系结构中行为无死锁等方面的问题.

组件的失配问题是指在软件的复用过程中,由于待复用组件对最终系统的体系结构和环境的假设与实际情况不符而导致的冲突.在基于软件体系结构组装阶段,失配问题主要包括:① 由组件引起的失配,包括由于系统与组件基础设施、组件控制模型和组件数据模型的假设存在冲突引起的失配;② 由连接子引起的失配,包括由于系统与组件交互协议、连接子数据模型的假设存在冲突引起的失配;③ 由于系统成分与全局体系结构的假设存在冲突引起的失配等.要解决失配原因,首先需要检测出失配原因,并在此基础上通过适当的手段消除失配问题.

1.1.4 异种组件模型的出现

CBSD以软件架构为组装蓝图,以可复用软件组件为组装模块,支持组装式软件的复用,大大提高了软件生产效率和软件质量[17].为此,国内外对于这

一技术的研究正在不断深入，同时大型的软件公司（例如 SUN，Microsoft）及软件组织机构（如 OMG）都推出了支持组件技术的软件平台. 其中具有代表性的实现级工业标准组件模型有 CORBA/CCM（CORBA component model）[18-20]，COM/DCOM[21]，EJB[22]. DCOM 主要用于私有的 Windows 平台，EJB 因 Java 语言的平台无关性已经成为目前主流的组件技术，CCM 因 CORBA 的语言、平台和操作系统的无关性而成为最具发展潜力的组件模型. COM/DCOM，CORBA/CCM，EJB 三足鼎立构成竞争和互操作并存的格局[23]. 同时研究性的组件模型有 Charles 大学的 SOFA/DCUP 组件模型[22]以及我国北京大学的青鸟组件模型[24]等.

在嵌入式领域，嵌入式构件技术也正成为研究的热点[31]. 例如 DESS 项目，该项目的主要研究目标就是基于构件的嵌入式实时软件开发. 而另一个相关项目 EMPRESS 项目的研究目标是提出新的嵌入式实时软件开发方法和工具，用于支持基于构件的嵌入式软件的动态进化（升级）. 目前嵌入式构件技术的主要研究内容有：① 嵌入式构件基础设施. 构件基础设施为软件构件提供了一个运行环境，这样的运行环境在许多构件模型中也称为构件容器. Völter 等人提出了一种基于代码生成技术的嵌入式构件基础设施开发的方法，该方法要求首先提供构件信息、系统配置信息和硬件设备信息来作为配置验证器（config validator）的输入，配置验证器根据这些信息来判断是否可以产生相应的构件容器，如果可以的话，由容器产生器（container generator）来产生相应的构件容器的源代码. ② 基于构件的嵌入式操作系统. 在当前普适计算（pervasive computing）的环境下，嵌入式应用日趋复杂. 嵌入式应用需求差异较大，这就要求作为系统支撑平台的嵌入式操作系统支持模块化开发，具有灵活的可配置能力，使得系统易于根据用户需求而定制为应用相关的操作系统. 然而传统的嵌入式操作系统往往是平台相关的、体系结构整体不可分（monolithic）的系统，开放性和系统重用性差，不能较好地满足需求. 为此，将构件技术应用于嵌入式操作系统中也成为一种可行的途径. ③ 嵌入式构件开发工具和集成开发环境. TAXYS 工具主要针对于实时通讯软件系统的设计与校验. 该工具将 ESTEREL 语言作为应用系统的开发语言，并采用时间自动机（timed automata）来捕捉系统的实时行为并建立相应的模型，同时利用 KRONOS 工具进行模型分析. ④ 嵌

入式构件模型. 当今国内外提出的比较著名的嵌入式构件模型有 PBO 模型、KOALA 模型、SEESCOA 模型和 PECOS 构件模型等.

1.1.5 企业应用集成

Web 为人类提供了一个全球范围内的分布式系统,在 Web 下实现不同组件产品的集成,将会使组件的应用更加广泛,更大范围的分布组件也可实现交互,这样就形成一个开放式的系统,它使用户能够透明地应用不同厂商制造的异构型计算资源. 因此,在软件组件复用技术研究成果和软件组件标准广泛被接受的情况下,利用开放的 WWW 环境,通过新的 Web 技术和规范,实现开放的应用,将具有极大的诱惑和广阔的应用前景[25]. 在互联网快速发展的今天,网络为人们的协作提供了广阔的平台,在 WWW 下实现不同对象的互操作性就越发迫切了. 当今,组件技术已经成为计算环境的基本组成之一,众多中间件产品和开发工具提供了对不同组件模型的支持,特别在分布式、企业级应用软件系统中,人们把软件的组件化作为解决维护、扩展和升级问题的惟一途径[26-27].

由于没有统一的组件标准,所以目前存在着不同的组件模型,这些异质的组件都有其各自的特点,包括不同的应用平台、不同的实现语言和不同的技术等. 在 Internet 环境中,在一个集成商业程序的所有参与者间确保单一、统一的体系架构是十分困难的. 组件技术的目标本来是为了能够更好地实现代码重用,但缺少异构组件间的互操作,用户就不得不为各种不同的分布式组件系统编写相同功能的代码[28]. 对企业应用来说,基于不同规范的异构分布式组件的互操作就成为一个非常重要的特性. 在制定合理的异构组件间互操作标准后,组件集成后就能得到一个一致的可运行系统. 因此,在 Web 下解决软件组件互操作性,实现异构组件的通信、协作,使得软件开发可在一个更加开放、更加宽松的 WWW 平台上,充分利用 Web 计算的特点来完成. 构造这样一个应用最大、最广泛的分布式系统正是软件应用发展的方向.

1.1.6 面向服务的计算

近年来,面向服务的体系结构 SOA(service-oriented architecture)受到学术界和工业界的广泛关注. 和面向对象体系结构不同,SOA 将应用程序的不同