摘 要

随着互联网经济的快速发展，以网络为中心的商业模式逐渐成为主导商业模式。这使得对基于 Web 的工作流系统的需求大量增加。虽然，目前已经有一些公司针对这种需求，在原有的工作流系统之上进行了改造，使其支持 Web 应用。但是传统工作流的集中控制体系结构，从根本上难以适应 Web 应用松散耦合，分散控制，以及灵活多变的商务过程。

正是在这样的背景下，本文提出了一个新的基于移动代理技术的工作流管理系统体系结构——协同工作引擎。协同工作引擎的核心技术包括工作流技术、移动代理技术和 Web 服务技术。移动代理本质上是一个应用程序，它能够自主移动并运行，完成事先定义的任务。移动代理具有以下优点：开放型、灵活性、健壮性、伸缩性，智能性等。正是由于有了这些优点，使得移动代理技术作为新兴的分布式计算和移动计算支撑技术，可以作为一个提供个性化服务的网络服务框架结构。

Web 服务技术指的是一系列技术和规范的集合。它是由 Microsoft 公司提出的基于互联网的开发模型。Web 服务以其良好的应用集成方案，很快得到了众多研究机构和公司的关注，其中 Microsoft 和 IBM 两家公司已经分别提出了各自的 Web 服务体系结构，并且成为目前工业界两种主流的体系结构。

基于以上理由，本文充分利用三种核心技术的各自的优势，将三者相结合优势互补。分别采用 Web 服务技术和移动代理技术作为分布式计算模型和分布式应用集成框架，从而解决工作流系统的底层运行的分布式计算环境和与其它 Web 应用集成标准问题。同时，本文还引进了本体技术进行流程语义描述，并提出了一个基于工作流的本体及其面向对象的设计方法。最后，基于 XML 技术本文详细论述了一种工作流建模方法。

由于协同工作引擎的设计，涉及多个研究领域交叉研究，要解决问题多而复杂，因此本文仅从几个问题入手进行研究和探索，提出一些个人见解。

Abstract

With the swift development of Internet, the commerce style focusing on network becomes the main stream style. Gradually. This makes work flow system based on Web increases greatly. Some companies had improved the existing work flow systems to support Web applications according to this requirement, but the traditional centralized controlling architecture can not adapt to the today's commerce process of being loose coupled, distributed controlling and flexible.

Under this background, in this thesis we point out one new work flow managing architecture-cooperative work engine which is based on Mobile Agent technology. The core technologies include work flow technology, Mobile Agent technology and Web Services technology. In essence , Mobile Agent is an application, and it can automatically migrate and execute to accomplish the desired task. Mobile Agent has following merit: being open, flexibility, robustness, extensibility and intelligence. For all these advantages, Mobile Agent had been the supporting technology for the emerging distributed and mobile computing technology, and can be used as the Web Services framework to provide characteristic services.

Web Services means the collection of several technologies and specification. It is pointed out by Microsoft Company as model for development based on Internet. For it's good application integration project, Web Services had won attention of many research institute and companies. The two Web Services architecture which pointed out by Microsoft and IBM respectively now had been two main stream architectures in industry.

Based on above reasons, in this thesis we used the advantages of three core technologies and made them complement with each other. We used Mobile Agent and Web Services as distributed computing model and distributed application integration framework to solve the bottom distributed computing environment of work flow system and to integrate with other Web applications. Simultaneously, we

pointed out the designing method of Ontology based on work flow. Finally, we detailed demonstrated one work flow modeling method.

Because the cooperative design means the cross research among many research domain, and the problems involved are many and complicated, so we only do some research and investigation from the point view of some problems and point some personal opinions.

Key words: Mobile Agent, Workflow, Web Services, Ontology.

独 创 性 声 明

本人声明所呈交的学位论文是本人在导师指导下进行的研究工作及取得的研究成果。据我所知，除了文中特别加以标注和致谢的地方外，论文中不包含其他人已经发表或撰写过的研究成果，也不包含为获得电子科技大学或其它教育机构的学位或证书而使用过的材料。与我一同工作的同志对本研究所做的任何贡献均已在论文中作了明确的说明并表示谢意。

签名：张在龙　　日期：2003年3月2日

关于论文使用授权的说明

本学位论文作者完全了解电子科技大学有关保留、使用学位论文的规定，有权保留并向国家有关部门或机构送交论文的复印件和磁盘，允许论文被查阅和借阅。本人授权电子科技大学可以将学位论文的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或扫描等复制手段保存、汇编学位论文。

（保密的学位论文在解密后应遵守此规定）

签名：张在龙　　导师签名：

日期：2003年　3月 2日

第一章 绪论

1.1 工作流技术研究的起源和发展

工作流技术的起源可以追溯到上个世纪八十年代初期。当时，PC 机尚未作为信息处理工具而出现的时候，纸张成为日常业务活动不可替代的载体。而这种古老的载体在信息的处理、组织、存储以及查询检索等方面都是很低效的。因此，人们希望出现一种无纸化、计算机使能的工作环境。一些公司建立了自己专用的或者可商品化的表单传递应用系统，通常运行在大型机或小型机上，用于实现日常表单处理的电子化与自动化。这种系统可以看成是现在工作流管理系统一个雏形。

到了二十世纪八十年代中期，FileNet、ViewStar 等公司率先开拓了工作流产品市场，成为最早的一批工作流产品供应商。他们把图像扫描、符合文档、结构化路由、实例跟踪、关键字检索以及光盘存储等功能结合在一起， 形成了一种全过程支持某些业务流程的集成化软件，这便是早期的工作流管理系统。比较典型的有 FileNet 于 1984 年推出的 Workflo Business System，ViewStar 于 1988 年推出的 ViewStar。很显然，这种增值性质的集成化软件为企业简化和重组自己的关键业务流程提供了一种非常合适的方案；而且，由此我们还可以看出，工作流从最初的诞生之日起就是作为一种面向过程的系统集成技术而出现的，只不过限于当时的计算机发展水平，它所集成的功能较为简单而已。

进入二十世纪九十年代，随着计算机的普及、网络的延伸，现代企业的信息资源越来越表现出一种异构、分布、松散耦合的特点，企业的分散性、决策指定的分散性、对日常业务活动的详尽信息的需求以及 Client/Server 体系结构、分布式处理技术（CORBA、WWW、OLE、JAVA）的日益成熟，都说明了实现大规模的异构分布式环境，使得相互关联的任务能够高效运转、并接受密切监控以成为一种趋势。在这样一个技术背景下，工作流管理系统也由最初的创建无纸办公环境，转而成为同化企业复杂信息环境、实现业务流程自动执行的必要工具。这样的一个转变，把工作流技术带入了一个崭新的发展阶段，使得人们从更深的层次、更广的领域上对工作流展开了研究。

工作流技术的标准化组织——工作流管理联盟（Workflow Management Coalition，WfMC）于 1993 年成立。它的成立标志着工作流技术在计算机应用研究领域被明确地划分了自己的一席之地，相应的概念与术语也得到了人们的承认。在全球范围内，对工作流的技术研究以及相关产品开发进入了更为繁荣的阶段，更多更新的技术被集成进来，文件管理系统、数据库、电子邮件、移动式计算、Web 服务等都已被容纳到工作流管理系统当中。

1．2　工作流技术的研究现状

1．2．1　工作流的有关定义

十几年来，不同的研究者对工作流分别提出了不同的定义。到目前为止，对于工作流仍没有完全统一的定义，下面我们列举几个具有代表意义的定义，它们分别从不同的角度对工作流概念进行了描述。

- 工作流管理联盟的定义

工作流是一类能够全部或部分自动执行的业务过程，根据一系列的过程规则，文档、信息或任务能够在不同的执行者之间传递、执行。

- Giga Group 的定义

工作流是经营过程中可运转的部分，包括任务的顺序以及由谁来执行它、支持任务的信息流、评价与控制任务的跟踪、报告机制。

- IBM Almaden Research Center 的定义

工作流是业务过程的一种计算机化的表示模型，定义了完成整个过程所需用的各种参数。这些参数包括对过程中每一个单独步骤的定义、步骤件的执行顺序、条件以及数据流的建立、每一步骤由谁负责以及每个活动所需要的应用程序。

- Amit Sheth 的定义

工作流是涉及到多任务协调执行的活动，这些任务分别由不同的处理实体完成。一项任务定义了需要做的某些工作，它可以以各种形式来进行定义，包括

在文件或电子邮件中的文本描述、一张表格、一条消息以及一个计算机程序。用来执行任务的处理实体可以是人，也可以是计算机系统。

- W.M.P. van der Aalst 的定义

  工作流是一系列工作的偏序集。工作的序列可以有多种方式。

以上这些对工作流的定义，是用非形式化语言对工作流所进行的描述，虽然各有不同，但基本上都达成了这样一个共识：工作流是业务过程的一个计算机实现，而工作流管理系统则是这一实现的软件环境。这些工作流的定义分别反映了业务过程如下几个方面的问题，即业务过程是什么（由哪些活动、任务组成，也就是结构上的定义）、怎么做（活动间的执行条件、规则以及所交互的信息，也就是控制流域信息流的定义）、由谁来做（人或者计算机应用程序，也就是组织角色的定义）、做的怎样（通过工作流管理系统进行监控）。

1．2．2　工作流模型

工作流模型是对工作流的抽象表示，也就是对业务过程的抽象表示。由于工作流需要在计算机环境下运行，因此建立相应的工作流模型就是必不可少的。工作流模型应该完整提出支持工作流定义的概念，为建模用户提供工作流定义所需要的组件或元素。理想的工作流模型能够清楚地定义任意情况下的工作流，能够适应用户在建模过程中提出的各种要求。然而，到目前为止，人们虽然提出了不少有意义、有见解的工作流模型，但从模型的能力上看，距这一理想情况尚有意定的距离。

由于一个工作流必需有限描述清楚一个业务过程是怎样进行的。因此，许多工作流模型都从过程定义入手，比如流程图、状态图、活动网络图等等。这一类基于有向图的模型的优点是比较直观，容易理解，一般情况下图中的节点表示活动或者状态，而有向弧则表示节点间的时序依赖关系。不少工作流产品正是采用了此种模型，但其缺点是比较简单，不能处理复杂的过程逻辑，缺乏柔性。

Winograd 与 Flores 在语言行为（Speech Act）理论的基础上提出了一种基于对话的工作流模型，这种工作流模型是在客户方与服务方在这两个角色之间的

语言行为交互上对工作流过程进行了定义：他们认为人的语言不仅能够用来描述事物、交流信息，而且还能够进行行为的计划与协调，即通过语言能够承诺自己未来的行为，通过语言也可以协调自己与他人的合作。基于语言行为理论的工作流模型是由一系列闭合的工作流环相互连接而成的，每个工作流环都被四个语言行为(Speech Acts)分为四个阶段，包括需求阶段、协商阶段、执行阶段与满意阶段。Action Technologies 的工作流产品 ActionFlow 就采用了这种工作流模型。

Petri 网也被用来建立工作流模型，Ellis 和 Nutt 在 Petri 网的基础上提出了 ICN (1nformation Control Nets)模型，它实际上是高级 Petri 网的一个引申，在这里库所表示活动，而变迁则表示活动间的转移； Van der Aalst 则在 Petri 网的基础上定义了 WF-net)，即工作流网，在工作流网中变迁被用来表示活动，而库所则表示活动的使能条件。Van der Aalst 还把工作流管理联盟在规范中提出的几种基本的工作流原语映射成为相应的 Petri 网模型，由此建立了工作流网的基本组件与触发机制。

除了以上这几类工作流模型，还有许多其他形式的工作流模型，比如活动树(Activity Tree)的模型，它是以一个树状结构来表达工作流过程的，从根节点开始，过程被逐层地分解为由各级子节点所代表的活动，而活动间的执行顺序则是由左至右逐个分支地进行。Andreas Geppert 等提出了 Broker / Services 模型，即代理朋/服务模型，它定义了较为精确与严格的形式化语义，用代理来表示工作流执行过程中的处理实体，用服务来表示所要执行的活动，代理的行为是采用 ECA(Event-Condition-Action)规则描述的。

由于工作流不仅仅需要明确地表达经营过程中的活动以及活动间的关系，而且还要对活动间所传递的信息、活动的执行实体、活动所需要的资源等等方面进行定义，因此，人们便在工作流模型中加入了描述数据、组织、资源的部分，比如工作流管理联盟就明确提出了工作流相关数据、工作流控制数据及工作流参与者、角色等概念。在很多工作流产品中也允许用户在一定范围内定义数据、人员等。为了使工作流模型在描述信息、组织与资源上的能力更强，人们逐渐把相关的描述部分扩充为一个个较为完整的模型来更有力地支持工作流的建模。比较典型的有 WIDE 项目中提出了由组织模型、信息模型与过程模型

这三个子模型共同组成的工作流模型。

1．2．3　工作流中的事务管理

事务的概念来自于数据库研究领域，用以解决数据的并发访问和出错恢复问题。事实上，工作流也可以看成是一系列有序操作的集合，只不过这些操作的对象具有更广的内涵，并不仅仅限于数据库中的数据。因此，工作流也同样具有事务特性。

人们首先研究了在数据库事务模型的基础上所提出的许多高级事务模型(Advanced Transaction Model)，包括嵌套事务模型、多层事务模型、Sagas、分支付正合事务模型、柔性事务模型、ACTA 等。高级事务模型通常把一系列的操作分组成为层次化的结构，并且放宽了经典事务模型对 ACID 特性的要求，以便适应不同性质的实际问题，因此又被称为扩展事务模型。不同的高级事务模型往往是针对不同的问题而提出的，有着不同的特点。

由于高级事务模型在解决长时间事务方面仍有很多局限性，人们把注意力由专门的数据库事务扩展到了工作流这一范围。德国 Stuttgart 大学的 ConTracts 研究项目提出了自己的高级数据模型和高级并发控制机制，已经具备了一定的工作流描述能力，而且从解决问题的思路来看，ConTracts 模型跳出了原有的高级事务模型的局限，他们认为：扩展原有的事务模型并不能解决问题，因为长时间的计算过程要比一个具有 ACID 特性的事务复杂得多，数量上的变化将导致事物本质的变化，宏观与微观的差距将使它们的一致性问题变得各具特色，决不能一概而论。

Amit Sheth 在对高级事务模型进行研究的基础上则提出了事务工作流(Transactional Workflow)的概念)，他认为，许多高级事务模型的执行结构都很有限，高级事务模型所预先定义的许多属性对于工作流应用而言可能并不必要：而且在工作流的执行过程中，有些参与执行的系统可能并不支持这些事务模型：另外，事务模型所注重的是保护数据的一致性，对于执行不同任务的相互独立的系统之间的协调则并不擅长。Amit Sheth 完全从工作流的角度提出了任务的结构化定义以及基于任务间依赖关系的工作流定义，还对系统的实现方法提出了有意义的见解。

1．3　目前工作流技术中所存在的不足

尽管经过工作流产品供应商与工作流研究人员十几年的不懈努力，使得工作流技术由最初的萌芽逐步发展起来，并取得了相当的成果，但是从工作流系统的实际应用状况来看，还远未达到人们所期待的普及状态。在经营过程中采用工作流管理系统的企业仍只是一少部分，而且这些系统的应用范围也很有限，并不能全方位地支持企业的关键业务流程。从企业用户应用的角度来分析产生这种状况的原因，主要有以下几点：

(1)工作流的运行必须要有底层的通讯基础结构的支持，但是就目前能够实现分布计算环境的产品来看，它们在实际应用中仍然显得不够成熟，在安全性、容错性、可靠性等方面均不能满足企业的需求，而且在价格上也给企业造成一定的负担。

(2)缺乏标准。不同的厂商所提供的工作流产品可能具有自己独立的一套工作流模型、工作流定义语言以及 API 函数。在这种缺乏标准的状况下，用户一旦选定一种产品之后，就很难再过渡到其他同类产品之上了，而且不同的系统之间缺乏互操作的接口。

尽管工作流管理联盟的成立有助于改善这种情况，但若想实现类似于关系型数据库这样的统一标准(比如关系数据模型、SQL 语言等)，仍有很长的路要走。

(3)实现的复杂性。工作流应用的开发不仅仅是过程的定义，还需要做其他许多任务，比如：对外部的应用系统进行封装，建立工作流机运行所必须的分布计算环境，设计开发相应的用户界面等等。就目前的工作流产品来看，几乎不能为这些任务提供什么有力的帮助，所有的工作流应用都需要工作流产品供应商与应用开发人员进行很长时间的合作才能最终完成。另外，工作流系统的实施给企业带来的不仅是技术上的变化，同时也会对企业原有的管理制度造成一定的影响，这也是对企业的一种冲击。

(4)工作流技术本身的不成熟性。目前尚没有一种工作流产品或原型系统能够在过程执行的可靠性与一致性方面达到与关系型数据库管理系统同水平的功能。尽管在实际应用时对工作流系统并不需要那样高的性能要求，但具备这样的能力对于一个工作流系统而言是很重要的，只有这样，才能使企业有信心采

用工作流技术来对那些应用其他技术(如数据库)实现的关键任务应用进行重组。

从我们对工作流的研究来看，我们认为工作流技术自身的不成熟性从较为根本的几个层次上来看主要表现在以下三个方面：

(1)在工作流的模型描述方面，缺乏一种支持过程定义、过程演进以及过程分析的形式化的数学模型。工作流模型的核心是对过程的定义，包括组成过程的基本活动以及活动之间的时序关系。目前的各种工作流模型，大部分都是从直观感觉出发，以图形语言或者文本语言来定义工作流过程，这种定义的方法实际上仅仅是处于用户层上，即对用户而言是比较理想的方式，但并不利于实际系统的实现，也无法对工作流的本质特征进行描述，更谈不上对过程的分析与评价。虽然有的模型具有形式化的数学描述，比如 WF-net)，但从模型能力上距离对工作流的本质描述仍有差距。由于缺乏理论层的模型支持，使得工作流在应用的许多关键特性上无法得到保证，包括柔性定义、过程重用、事务管理、异常处理等，这都大大限制了工作流在企业应用中的推广。

(2)在工作流的执行方面，缺乏一个标准化的集成框架来支持对企业常用的分布式应用的集成。企业在应用工作流进行业务流程的运作时，最为关心的就是工作流系统能否与企业原有的各个应用系统(比如 MRPII 系统、CAx 系统等)很好地集成起来，使它们成为一个完整业务流程当中的有机组成部分，而不是象原来那样处于一种“孤岛”状态。但目前来看，工作流应用中的一个很大的瓶颈就是工作流管理系统所能支持的企业应用太少，在集成的方式方法上没有统一标准，很大程度上要受到外部应用的限制。因此，在工作流系统与企业应用间亟待建立一个性能良好的“粘合层”，最好是独立于不同企业应用的一个标准的集成框架，这将极大地提高工作流系统对企业应用的适应性。

(3)在工作流的仿真评价方面，尚处于一种几近空白的状态。应该说，在缺乏仿真方法与仿真工具支持的情况下，整个工作流系统是不完善的，因为人们难以预料所布署的工作流过程将有可能出现怎样的结果，它有哪些不合理的地方，它的性能指标如何，这一切都必须等到实际运行以后才能由工作流管理系统所记录的数据中获得，显然，这并非是一种合理的方式。针对工作流进行仿真的难点主要在于：仿真的性能指标不好确定：仿真的内容较为复杂等。

以上提到的工作流技术面临的各种问题，都导致了目前工作流产品虽然不

少，但真正适用并起到良好效果的并不多。

1．4　工作流技术发展方向

工作流技术发展到今天，传统的技术手段、体系结构和相关标准已经越来越不适应复杂多变的情况，而现实的应用需求越发显得紧迫，这一切都促使工作流技术不得不进行一场技术革命。这场技术革命的特征就是打破研究领域的界限，进行一次新的技术重组，就像当年工作流技术的研究从其它研究独立出来一样。新的工作流集成框架涵盖了更加广泛的领域研究技术，如人工智能、分布式计算、Web Services、知识工程等等。

下面我们分别从市场、技术、理论三方面，对于工作流技术的未来发展趋势进行概括：

从工作流市场的发展来看，主要有三个特点：(1)市场潜力大，仍将保持良好的增长势头：(2)工作流产品的价格将不断下降，这是竞争与普及双重作用的结果：(3)产品的应用领域逐步由通用走向专用，具有行业特点的工作流产品将占领市场，比如工程、制造、电讯等领域。

从理论的发展来看，虽然它在一定程度上滞后于应用，但是在迫切的应用需求的驱动下，也必然会逐步完善。几个较为突出的发展方向有工作流的形式化描述、工作流的事务模型、工作流的设计与分析方法等。

从技术发展来看，随着工作流这一集成框架下所容纳技术的不断拓展与成熟，工作流系统将成为企业信息环境中不可缺少的软件平台，Thomas Koulopoulos 预言工作流系统将最终成为覆盖于各类台式机与网络操作系统(如 Windows、Unix、WindowsNT)之上的业务操作系统 BOS(Business Operating System)，它将带来操作系统的一次革命。在这一基础上，我们还可以进一步预言，在家电信息化逐步成熟的将来，工作流极有可能从经济生活走入家庭生活，为人们的日常生活定制理想的自动流程，成为新时代的家庭信息平台 FIP(Family Information Platform)。

另外，工作流与移动代理技术相结合也是目前研究的热点。可移动代理具有许多优点，比如在一定条件下能够减少网络流量、适合于移动用户、有利于

数据集成、具有并行机制等，因此很适用于工作流管理系统的构建：企业的每一个经营过程的实例可以由一个移动代理来处理，代理在预先定义好的步骤下在分布的网络节点上执行，当代理移动时，它携带着过程所需的执行代码与数据，无需每一步都通过中央的数据库服务器来交换数据。

此外，从应用角度出发，工作流技术与 Web 服务技术相结合正在成为基于 Web 应用典型解决方案，在这种解决方案中分发挥了工作流系统的调度能力、对业务过程建模能力和 Web Services 的良好的应用集成框架。

本文致力于工作流技术、移动代理技术和 Web Services 技术三者相结合。分别采用 Web Services 技术和移动代理技术作为分布式计算模型和分布式应用集成框架，从而解决工作流系统的底层运行的分布式计算环境和与其它 Web 应用集成标准问题。而代理的运行环境就是工作流的运行环境。由于代理的交互是基于本体（ontologies）的。因此，利用本体对工作流过程进行定义就是自然的了。这样做同时可以解决工作流模型的形式化描述的问题。而本体对语义信息的描述能力也进一步增强了工作流系统交互能力。

在下一章我们将详细介绍移动代理技术和 Web 服务技术的相关背景知识。

1．5　工作流术语

为了便于在本文后面的章节中展开论述，我们首先给出将要用到的一些概念和术语的定义，这些概念和术语在本文范围内将保存一致。

- 任务（Task）

是工作流过程的一个原子工作条目。任务可以由一个 Web 服务实现，更复杂的情况下，可能有多个 Web 服务共同完成一个任务。

- 参与者（Actor）或资源（Resource）

是一个人或机器，通过履行一定的服务职能来完成特定任务。在本文范围内，参与者还可能是一个软件代理。

- 角色（Role）

是一个任务集合抽象。角色表示一个工作流参与者在整个过程中完成任务。

对于每一个这样的任务相应的角色代理（Role Agent）调用相应的 Web 服务来完成。

- 过程（Process）

是一个用机器可以自动处理的方式进行表示的业务过程描述，通常由一个按照一定的顺序执行的任务列表组成。

- 工作流实例（Workflow Instance）

是指运行期间的工作流过程。它与执行该过程的特定资源（Resource）绑定在一起。

1．6　论文组织

本文共分 6 章，其余部分的组织如下：

第二章，工作流集成与互操作。在这一章我们将重点解决工作流的集成和互操作问题。

第三章，协同工作引擎的设计。我们在这一章中首先分析了传统工作流管理系统中存在的问题及产生这些问题的根源。并提出了一个基于移动代理的解决方案和一个新的工作流管理系统的体系结构及其工作原理。

第四章，工作流建模。本章以一个在线服务为例，说明了工作流各个动静态视图和基于工作流的数据库模型的构建方法。

第五章，协同工作引擎的代理设计。本章介绍了基于 IBM Aglets 的代理设计原型，以及部分实验结果。

第六章，结束语。对整个课题研究的重点，尚未解决的问题做出了总结。

第二章 工作流集成与互操作

在第一章绪论中，我们详细论述了工作流技术研究的起源、发展现状、存在的问题以及未来的发展方向等。在对现今工作流技术中存在的问题的分析当中，我们将工作流技术自身的不成熟性归结为三个方面原因：

1. 缺乏一个形式化的工作流描述模型。
2. 缺乏一个标准化的集成框架作为支持分布式应用集成的工作流系统的底层运行环境。
3. 缺乏工作流仿真评价方法和工具。

基于从市场需求、理论分析和技术发展等不同层面，对工作流技术今后的发展趋势作了简单分析，我们认为工作流技术与 Web 服务技术和移动代理技术的结合是今后工作流技术发展主要趋势。

本章中，我们将详细论述基于 Web 服务框架的应用集成和基于移动代理技术的互操作。

2.1 基于 Web 服务应用集成

当今世界正在经历着从以人与人之间直接交互为基础的经济时代向基于软件服务之间的交互的经济时代的变迁。这种变化为提高生产力和缩短生产流程周期方面带来了巨大的进步。在电子商务时代，人们只需要决定做什么，而软件程序会根据人的意愿决定如何做才是最佳解决问题的方法。这里关键的技术，就是 Web 服务技术及相关标准的出现。今天，从不同的组织获得服务来进行基于互联网的交互还是一件很困难的事。这里，最主要的原因是由于不同的服务提供者采用自己特有的方法来提供解决问题的方法。而采用标准的方法，进行 Web 服务的定义使得发布、操纵和使用合成服务变得很容易。Web 服务标准允许我们将 Web 上所有的应用作为一个个的 Web 服务链接起来，我们可以通过互联网装配它们来解决我们自己的问题。接下来，我们就来介绍一下有关 Web 服务基本概念和基于 Web 服务应用集成体系结构。

2．1．1　Web服务的概念

我们首先区分两个术语：Web Services和Web Service。前者，Web Services是指整个架构Web服务的技术框架；后者，Web Service是使用Web Services而架构出来的Web服务实例。分别译为Web服务技术和Web服务。

那么，什么是Web服务呢？从表面上看，Web服务就是一个应用程序，它向外界暴露出一个能够通过Web进行调用的API。这就是说，你能够用编程的方法通过Web来调用这个应用程序。我们把调用这个Web服务的应用程序叫做客户。从外部使用者的角度看，Web服务是一种部署在Web上的对象（Web Object），它具有以下特征：

- 完好的封装性

Web服务既然是一种部署在Web上的对象，自然具备对象的良好封装性，对于使用者而言，他能且仅能看到该对象提供的功能列表。

- 松散耦合

这一特征也是源于对象/组件技术，当一个Web服务的实现发生变更的时候，调用者是不会感到这一点的，对于调用者来说，只要Web服务的调用界面不变，Web服务的实现任何变更对他们来说都是透明的，甚至是当Web服务的实现平台从J2EE迁移到了.NET或者是相反的迁移流程，用户都可以对此一无所知。对于松散耦合而言，尤其是在Internet环境下的Web服务而言，需要有一种适合Internet环境的消息交换协议。而XML/SOAP正是目前最为适合的消息交换协议。

- 使用标准协议规范

这一特征从对象而来，但相比一般对象其界面规范更加规范化和易于机器理解。首先，作为Web服务，对象界面所提供的功能应当使用标准的描述语言来描述(比如WSDL)；其次，由标准描述语言描述的服务界面应当是能够被发现的，因此这一描述文档需要被存储在私有的或公共的注册库里面。同时，使用标准描述语言描述的使用协约将不仅仅是服务界面，它将被延伸到Web服务的聚合、跨Web服务的事务、工作流等，而这些又都需要服务质量(QoS)的保障。其次，

我们知道安全机制对于松散耦合的对象环境的重要性，因此我们需要对诸如授权认证、数据完整性(比如签名机制)、消息源认证以及事务的不可否认性等运用规范的方法来描述、传输和交换。最后，在所有层次的处理都应当是可管理的，因此需要对管理协约运用同样的机制。

作为 Web 服务，其所有公共的协约完全需要使用开放的标准协议进行描述、传输和交换。这些标准协议具有完全免费的规范，以便由任意方进行实现。一般而言，绝大多数规范将最终有 W3C 或 OASIS 作为最终版本的发布方和维护方。

- 高度可集成能力

由于 Web 服务采取简单的、易理解的标准 Web 协议作为组件界面描述和协同描述规范，完全屏蔽了不同软件平台的差异，无论是 CORBA、DCOM 还是 EJB 都可以通过这一种标准的协议进行互操作，实现了在当前环境下最高的可集成性。

正因为 Web 服务技术具有以上这些特性和优点，使得 Web 服务技术作为一种 Web 应用集成框架，越来越受到学术界和工业界的重视。

Web Services 平台是一套标准，它定义了应用程序如何在 Web 上实现互操作性。你可以用任何你喜欢的语言，在任何你喜欢的平台上写 Web 服务，只要我们可以通过 Web 服务标准对这些服务进行查询和访问。

Web 服务的一个主要思想，就是未来的应用将由一组应用了网络的服务组合而成。只要两个等同的服务使用统一标准和中性的方法在网络上宣传自己，那么从理论上说，一个应用程序就可以根据价格或者性能的标准，从两个彼此竞争的服务之中选出一个。除此之外，一些服务允许在机器之间复制，因而可以通过把有用的服务复制到本地储存库，来提高允许运行在特定的计算机（群）上的应用程序的性能。

Web 服务技术的体系结构是面向对象分析与设计(OOAD)的一种合理发展，同时也是电子商务解决方案中，面向体系结构、设计、实现与部署而采用的组件化的合理发展。这两种方式在复杂的大型系统中经受住了考验。和面向对象系统一样，封装、消息传递、动态绑定、服务描述和查询也是 Web 服务技术中

的基本概念，而且，Web 服务技术另外一个基本概念就是：所有东西都是服务，这些服务发布一个 API 供网络中的其他服务使用，并且封装了实现细节。

2．1．2　基于 Web 服务的应用集成体系结构

我们已经知道，Web 服务技术是为解决在 Internet 环境下，松散耦合的 Web 服务之间进行互相调用、互相集成而设计的技术框架。以 XML、SOAP、WSDL 和 UDDI 为主干的 Web 服务技术赋予了 Web 服务一个与传统对象调用技术相似但又不太相同的体系架构。

Web Services 的体系结构是一种面向服务的体系结构（SOA）。

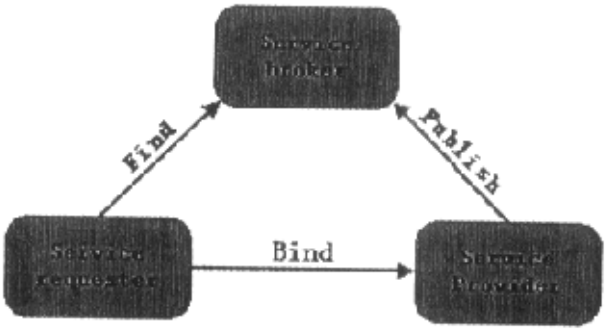

图 2-1　面向服务的体系结构

如图 2-1 所示，SOA 结构中共有三种角色：

- Service provider：发布自己的服务，并且对使用自身服务的请求进行响应
- Service broker：注册已经发布的 Service provider，对其进行分类，并提供搜索服务
- Service requester：利用 Service broker 查找所需的服务，然后使用该服务

SOA 体系结构中的组件必须具有上述一种或多种角色。

在这些角色之间使用了三种操作：

- publish操作：使Service provider可以向Service broker注册自己的功能及访问接口
- find操作：使Service requester可以通过Service broker查找特定种类的服务
- bind操作：使Service requester能够真正使用Service provider

为支持结构中的三种操作（publish、find和bind），SOA需要对服务进行一定的描述，这种服务描述（Service Description）应具有下面几个重要特点：首先，它要声明Service provider的语义特征。Service broker使用语义特征将Service provider进行分类，以帮助具体服务的查找。Service requester根据语义特征来匹配那些满足要求的Service provider。（因此，语义特征中重要的一点就是对Service provider的分类。）其次，服务描述应该声明接口特征，以访问特定的服务。最后，服务描述还应声明各种非功能特征，如安全要求，事务要求，使用Service provider的费用等等。接口特征和非功能特征也可以用来帮助Service requester对Service provider的查找。

注意，服务描述和服务实现是分离的，这使得Service requester可以在Service provider的一个具体实现（implementation）正处于开发阶段、部署阶段或完成（execution）阶段时，对其（具体实现）进行绑定。

另外，SOA中的组件相互之间必须能够进行交互，才能进行上述三种操作。所以Web Services体系结构的另一个基本原则就是使用标准的技术，包括服务描述、通讯协议以及数据格式等。这样一来，开发者就可以开发出平台独立、编程语言独立的Web Services，从而能够充分利用现有的软硬件资源和人力资源。

最后，SOA体系结构没有对Web Service的粒度进行限制，因此一个Web Service即可以是一个组件，该组件必须和其他组件结合才能进行完整的业务处理；Web Service也可以是一个应用程序。

2．2　基于移动代理的互操作

2．2．1　移动代理的概念

首先，我们给出一个移动代理定义。

定义：移动代理本质上是具有跨平台持续运行，能够在网络各主机间进行自主移动，自发地实现其功能，并与相关代理通讯的软件实体。

移动代理技术是分布式计算技术和人工智能技术完善发展基础上的产物。移动代理可以说是分布式计算技术的一个分支，但已经是用人工智能技术对分布式计算进行飞跃式改良后的成果。分布式计算的关键思想在于通过远程过程调用或对象引用实现跨平台的互操作，这种模式适合传统的相当稳定的网络环境和应用程序，但以目前及未来这种爆炸性增长的基于网络的信息资源和商务事务量，以及移动计算等新兴应用领域面前缺乏智能、主动和动态的信息处理能力而显得力不从心。

移动代理可以很好的满足上述的需求。移动代理本质上是一种可以从网络上的一台主机移动到另一台主机的代码或程序，并且可以自主地选择移动的时间和地点，因此移动代理也被称为移动智能体。代理在移动的过程中，它的自身状态被保存，并封装成信息传送到新的主机上，从而在新的主机上继续执行。所以，对于很多应用系统来说，移动代理是一个行之有效的选择：在客户机/服务器体系中它可以明显改善延迟和提高网络带宽利用率；在网络状况不佳时，还可以降低通信中断线的概率。

2．2．2　移动代理研究发展现状

移动代理研究主要可以分为以下七个方面：移动代理功能研究，移动代理系统研究，移动代理形式化研究，移动代理系统实现研究，移动代理应用研究，移动代理标准化研究，移动代理与其他技术整合的研究等。以下本文将从移动代理研究的各个方面具体探讨所取得的进展和存在的问题。

- 移动代理功能研究主要有：命名和定位，迁移，通讯，协调移动代理组。其中移动代理组的研究主要是针对一组移动代理而展开，如组模型，组通讯等。

- 移动代理系统研究主要有：安全，性能，容错，孤子检测与终止，调试与校验，资源分配与调度等。
- 移动代理系统实现研究主要有：移动代理编程语言，移动代理服务器体系结构，移动代理系统开发等。
- 移动代理应用研究分布的领域极其广泛。目前大都还是预研性的研究。在远程实时控制，移动客户支持，信息查询和分布数据访问，电子商务，网络管理和控制，供应链工作流和虚拟企业，网络入侵检测，软件或组件发布，QoS，分布计算和移动计算环境及应用等领域都取得了不少成果。
- 移动代理形式化研究主要有：移动安全等的形式化描述移动代理语言的类型系统访问控制的类型系统等。
- 移动代理标准化研究主要是标准化组织或团体提出移动代理研究中的一些技术标准和规范。
- 移动代理与其他技术整合的研究属于交叉性研究，目前主要有：与代理研究中其他领域技术的结合，与其他互联网技术的结合，与事务处理技术的结合等。

正是由于移动代理技术在计算模型和应用背景上都有很好的发展前景，所以学术界和工业界对移动代理都呈现出浓厚的兴趣，纷纷进行研究特别是 1994 年 Java 的推出。Java 作为一种网络语言，更是极大地推动了移动代理系统的开发。

国际上至今已涌现出 70 多个移动代理系统。其中有业界推出的 Telescript，Aglet，Concordia，Bee-agent 等，更多的是学术界推出的原型系统，如 Dagent，Mole，Ajanta，ARA，TACOMA，Messenger 等。我国南大新软件技术研究所也做了一些工作，但是国内目前还没有自己的移动代理系统。

在这些系统的基础上，开展了范围广泛的移动代理应用研究。例如，Telescript 系统作为 AT&T 产品 Personallink 的软件系统，为客户提供个性化的网络服务。TACOMA 在支持移动客户如 GSM，PDA 等方面做了一些研究，例如天气和股票报警等。Messenger 在分布科学计算如分布心血管建模和仿真等方

面做了一些研究，Dagent 在分布信息检索的研究，MAP 在网络管理方面的研究等。

2．2．3　移动代理技术特点及其优势

基于移动代理的分布式计算技术，打破了传统的客户/服务器模式诸多限制，提供了很多先进的特性和优点，为实现基于 Internet 的分布式计算开辟了新的技术路线。

这些新的特性包括：

- 动态执行方法：通过网络将代理动态地移动到目标主机执行，直接访问资源，从而避免了大量数据的网络传输，节约网络带宽资源，降低了系统对网络带宽的需求。
- 异步计算：移动代理的自主性使其不需要统一的调度，也不需要系统保持与网络长期的稳定连接，这对于构建基于因特网的应用具有重要意义。移动代理可以异步地在不同界点上运行，待任务完成后再将结果传送给用户，所以移动代理系统可以在低带宽、连接不是很稳定的网络环境中依然能够保持稳定的工作。
- 并行求解：为完成某项任务，用户可以创建多个代理，同时在相同或不同的节点上运行。移动代理的这一特性可将单一节点的负荷分散到网络的多各节点上，实现处理大规模、复杂问题的能力。
- 智能化路由：移动代理具有根据任务目标、网络通讯能力和服务器负载等因素动态规划下一步操作的能力。智能化路由能够很好地优化网络和计算资源，实现负载均衡，提高问题求解速度，避免对资源的盲目访问。

2．2．4　基于移动代理的工作流互操作的实现

基于移动代理技术实现工作流互操作的方法有：

1) 实现一个多代理系统作为工作流管理系统。此时，工作流管理系统可以看作一个基于移动代理技术的应用。这样，工作流之间的互操作可以用代理之间

的互操作来实现。

2) 利用移动代理作为一个建模技术来表示过程元素，如过程实例，活动实例等。这种方法，以移动代理技术作为关键的底层技术构建复杂的工作流引擎。本质上讲，这里也是一个多代理系统，不过相对于第一种情况，它更加紧密地结合了工作流和移动代理技术。

3) 不直接采用移动代理处理过程，而是作为一种机制。这里引入移动代理仅仅作为一种包装器，工作流管理系统之间的互操作是通过 WfMC APIs 实现的。

2.3 代理与工作流

移动代理技术的迅速发展使其成为分布式计算研究的重要分支之一，它克服了传统 C/S 模型的弊端，能够提供更加复杂的服务以及方面的进行扩充。为了获得比 C/S 模型更好的性能和资源管理能力，一些有关代理结构的问题，如直接服务，代理定位搜索机制等已经被考虑进来。

工作流技术，也是一个研究领域。与传统的办公自动化系统不同，它可以提供更好的柔性和可扩展性。有些研究者认为工作流系统就是一个天然的移动代理应用，因为在许多工作流系统中一些瘦客户端如 PDA，手机通过无线上网的设备被使用。许多移动代理特性可以很好的满足工作流系统的应用需求。离线的工作方式，使得用户可以在任何完成任务时候恢复网络连接。另外，工作流系统可以很容易的在异构平台上实现，这对于工作流系统集成和交互有着重要意义。

移动代理本质上就是一个软件组件程序，它能够在网络资源节点上实现自主移动，完成计算任务。它能够执行的任务包括，在线购物、实时设备控制和分布式科学计算等。

移动代理非常适合工作流管理系统，它能够按照工作流过程定义的每一步的规定在不同的站点上收集和处理信息，就像一个人到不同的部门办理相应的手续完成一个工作流程一样。不同的是，你再也不用真的去不同部门办公了，你要做的仅仅启动一个工作流程，而让你的软件代理去完成所有的事情。在这个过程当中，代理携带过程定义代码和相关数据，因而不需要和中央数据库通讯，也不需要源机器的支持。使用移动代理技术构建的工作流系统对现实世界

进行建模可以以一种更加直观形象的方式描述现实世界。

2.4 代理与 Web 服务

Web 服务具有很好的封装性，对于调用者而言它就是一个在 Web 上的服务对象，并且服务的描述和服务实现是分离的，因此，理论上我们可以采用任何一种语言的平台来实现 Web 服务，只要它的调用接口服务 Web 服务技术的规范。这就为代理技术与 Web 服务技术的结合创造了一个天然的条件，我们可以用一个代理来实现 Web 服务的功能，通过代理之间的交互完成服务调用。

此外，代理技术的引入为 Web 服务技术提供了很多新的特性：

Web Services 缺乏自学习性，它只知道自己的信息，不了解它的客户和顾客的能力信息。而代理具有自学习能力，通过学习和建模会不断的获知邻近环境中代理的最新能力。如果 Web Services 具有这一能力，就可以向 Registry 注册自己的最新能力，而某个潜在的客户就可以获得更好的服务；

Web Services 不象移动代理那样以本体作为相互通信的核心推理引擎，它没有充分考虑到如果服务请求者和服务提供者采用不同的本体，服务调用的返回结果对请求者来说是不可理解的，如果采用本体，可以增加核心逻辑转换功能来解决这一问题；

Web Services 的通信是“消息触发式”的被动通信，而且通信内容相对固定和简单，缺少推理性。当具有新消息时代理可以提供通知和刷新。而目前 Web Services 还不具备向注册中心请求刷新服务的能力；

现阶段下定义和使用的 Web Services 不具有自治性，而自治性是代理的一个重要特征，也是将来许多基于 Internet 的应用的特征。在代理社会中，一个代理可以感知周围对等实体，在必要时和其他实体进行协作，具有社会性，但在某种条件下单个代理也可以体现出独立性。本质上自治能力与个体间的协作和承诺是有冲突的，要想与其他代理协作或遵守自身的承诺，一个代理必须放弃一部分它的自治能力，然而在具有社会性和“责任感”的同时，一个代理仍然可以具有自治性，在首先尽可能确保对其他代理的承诺和协作的同时，它会适当的表现出自身的自治能力。在 Web Services 体系中，由于集中注册机制的完善，

这种自治能力被大大削弱了；

代理可以相互协作，形成一个软件意义上的团队和同盟，以提供更高级别的和更复杂的服务。当前的 Web Services 标准没有提供这样的“组合服务”。

2.5 本章小结

本章中，重点讨论了基于 Web 服务的应用集成和基于移动代理的互操作。

第三章 协同工作引擎的设计

3.1 传统工作流管理系统存在的不足

在第一章中，我们论述了现存工作流系统存在的不足。现在我们进一步分析一下存在的问题及其原因。

根据工作流管理联盟（WfMC）的定义，工作流系统的主要任务是公司处理跨部门的商业过程。然而，WfMC 提出的工作流管理模型在处理基于 Internet 的电子商务过程时存在以下不足之处：

（1） 缺乏支持松散耦合的体系结构。在基于 Internet 的电子商务的商务过程中，可能会包括很多来自世界各地的组织和公司，他们仅仅是为了某次商务活动而临时组成合作关系，一旦商务过程结束他们的合作关系也随之结束。他们的这种松散耦合的合作方式不同于某些合作伙伴之间紧密关系，他们的合作关系是随时都可能会变化或解散，这取决于他们各自的利益考虑。而 WfMC 的工作流管理系统模型是一种集中控制的系统结构，难以满足这种工作流应用的需求。

（2） 缺乏通用的通讯框架。不同的公司可能会采用不同的通讯平台，系统平台，甚至不同的工作流管理软件。这使得开发基于工作流的应用以及异构工作流系统之间的交互变得很困难。

（3） 缺乏本地自治的支持。在 WfMC 的工作流管理模型中，工作流过程被提前定义和部署好。过程参与者在工作流运行时几乎没有什么自治功能。这在一个公司或组织内部是可以接受的，然而对于那些来自因特网的不同公司来说出于对自身利益的考虑，他们需要这种本地自治能力好在某些情况下作例外处理，例如终止合作等。

尽管一些研究机构已经开发着手解决上述问题，例如统一接口规范和规范工作流开发过程。然而，松散合作的结构问题还没有得到很好的解决。本文提出了一个基于移动代理的工作流管理系统结构模型，可以很好克服上面提到的不足，更加适应基于 Internet 的电子商务应用。

3．2 基于代理的解决方案

3．2．1 移动代理

移动代理技术起源于分布式 AI，并很快扩展到分布式计算领域，成为分布式计算技术的一个分支。一个移动代理就是一个活动的自治计算实体，它携带代码、数据和执行状态。移动代理最主要的特性是自治、互操作和移动。移动代理能够自主移动和与环境互动。它也能够与环境或其他实体，如人，机器和其它代理进行通讯。

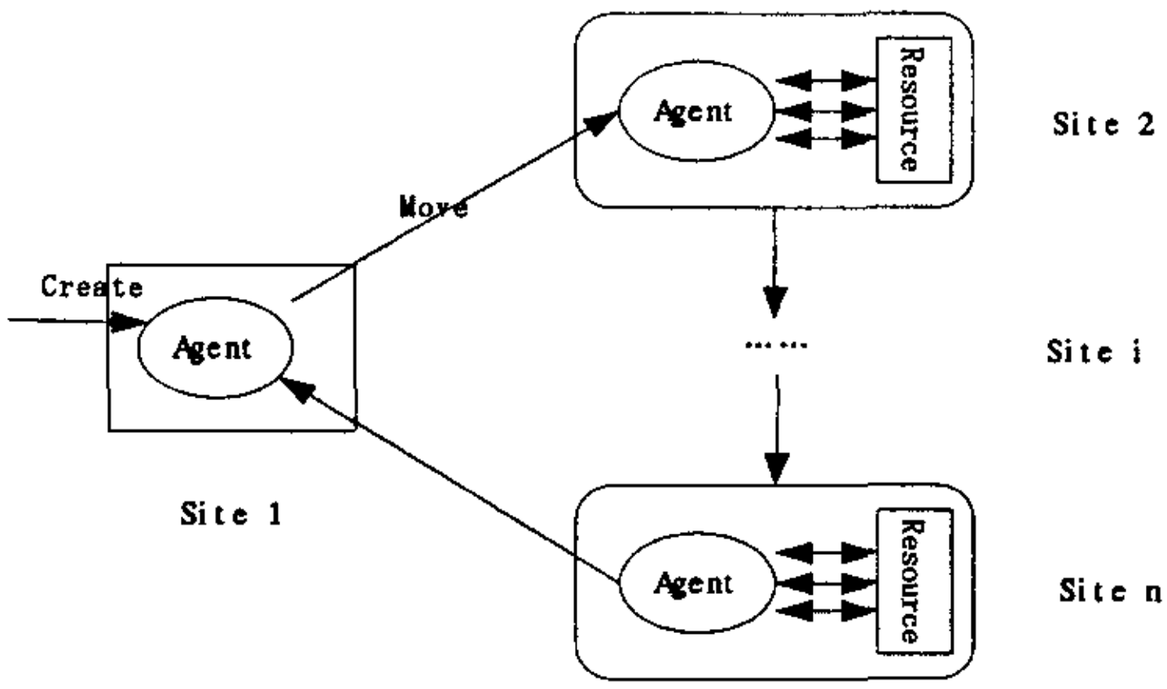

图 3-1　移动代理分布式计算模型

移动代理的计算模型如图 3-1 所示，与操作系统级的过程迁移不同，移动代理能够实现自主迁移。与远程过程调用也不一样，因为移动代理是能够自主活动实体，而且当它进行迁移时将同时携带它的执行状态信息。

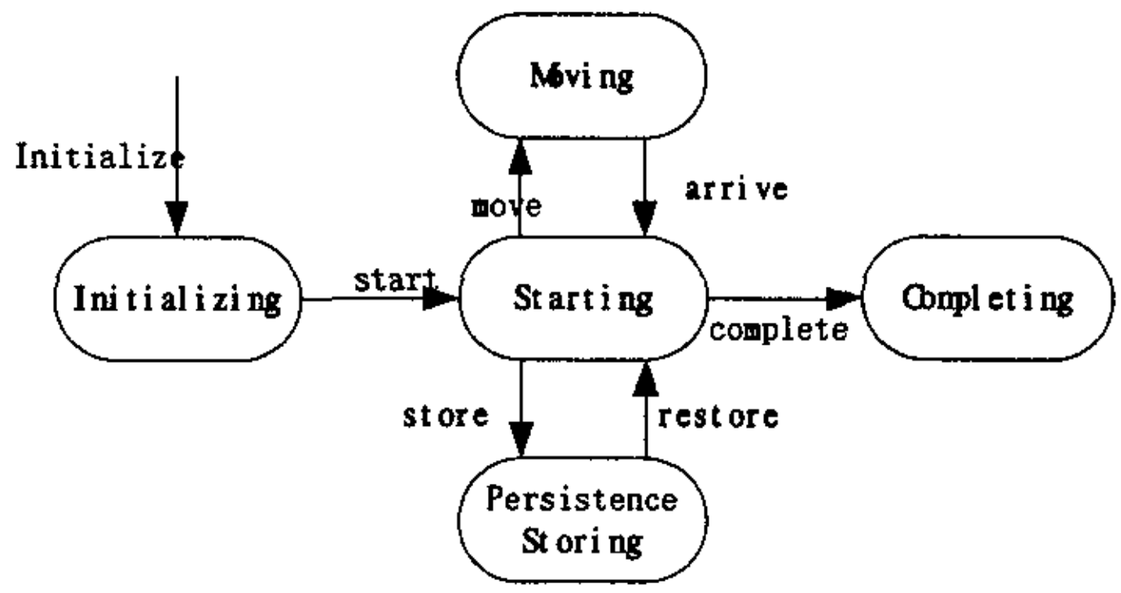

图 3-2　移动代理的生命周期模型

移动代理的生命周期模型如图 3-2 所示，它定义了在一个代理生命周期中的各种状态及其相互转化关系。生命周期模型清晰地刻画了移动代理分布式计算的形式。

毋庸质疑，代理技术的出现为解决工作流管理系统存在的问题带来了全新的思路。本质上，现存的工作流管理系统的问题都是由于它的集中控制体系结构所引起的。因此，我们就从问题的本原下手，利用代理技术改造工作流管理系统的体系结构（关于这个新的体系结构的详细内容，我们将在下一小节中做详细介绍）。本质上，这个新的体系结构是一个多代理系统结构。这里我们充分利用了代理的以下特性：

- 可移动性

移动代理在运行过程中，为了完成特定的任务可以从网络中的一个节点迁移到网络中的另一个节点运行。而传统的移动计算受到处理能力网络连接质量和代价、电源及安全性方面的限制；对于缓冲管理、资源的分配和回收等运用传统方法难以提供有效的解决方案。 因此移动代理避开了传统移动计算中的网络通信和处理能力的瓶颈，将交互与信息处理转移到具有很强处理能力和安全的主机上执行，对于移动计算技术的进一步发展颇具意义。

- 分布并行性(自治与协作)

在工作流管理系统中，可以将一个过程实例中的内容分解成若干任务，然后为每个任务创建一个任务代理完成相应的工作，每个任务代理则可以根据不同任务的具体情况迁移到适当的网络节点上并行运行，共同完成同一个任务。在运行过程中，各个移动代理之间可能是对等的，每个移动代理作为一个自治系统，相互协作，因此，这些运行的移动代理就构成了一个分布式工作流系统。

- 异步性

移动代理提供不同时间和空间范围内的互操作机制。传统的分布式计算一般基于同步方式，只有少数应用程序支持有限的异步交互（如电子邮件）。而移动代理引入了完整的异步计算环境，用户创建的移动代理可以异步地与处于其它时间和空间范围的主机交互，任务完成后将运算结果返回给创建者。由于，目前的 Internet 的网络性能和稳定性都比较差，因此代理的这一特性对于基于

Internet 的网络应用来说，具有重要的意义。

3．2．2 基于移动代理的工作流系统构建方法

我们知道一个软件代理实际上就是一个软件组件程序，它通过移动到资源节点上并与之进行交互完成运算任务。移动代理技术有很多优越的特性，可以极大提高现存工作流管理系统的性能。下面我们给出基于代理的工作流管理系统的构建方法。

1) 过程逻辑的封装

将业务过程逻辑和执行的状态封装在代理内，工作流活动的执行是通过代理向资源移动，并与之交互完成。

2) 构建代理控制中心

对工作流的控制，实际上就是控制代理执行活动。对代理的操作包括创建、调用、释放、挂起、唤醒、移动和执行。实现对代理的整个生命周期的服务需要 7 个基本组件的支持，即代理构建器、代理存储器、代理调度器、组件服务器、例外处理知识库、用户控制接口和事件服务器。

3) 建模与概念化

工作流策略是工作流系统的核心概念。通过对业务过程的分析和建模生成过程定义，每个过程定义都有一个工作流策略与之相关联。工作流策略中定义了与工作流过程有关的角色，活动起始和结束条件等。

3．3 协同工作引擎体系结构及其工作原理

在上一章中，我们论述了 Web 服务技术体系结构为应用集成带来了极大的便利，但同时由于其缺乏自治能力、缺乏推理能力，无法完成服务的自动发现、自动调用，自动合成和互操作，以及自动的执行监控，使其无法满足高级和复杂服务的需要。另外，传统工作流管理系统的刻板性，又使得无法完成柔性工作流的需求。而代理技术的出现，给上面两种技术的面临的问题提供了新的解决途径。

我们希望能构建一个新的工作流管理系统，它在体系结构的设计上，融合了 Web 服务技术、工作流和代理技术，并充分发挥了代理的自治和推理能力给与其它两种技术的弥补作用。它以移动代理技术作为分布式计算的模式，以工作流技术和规范作为底层基础设施，以 Web 服务的形式向上层用于提供服务，这就是本文提出的一个基于代理的协同工作引擎（Cooperative Work Engine，CWE）。

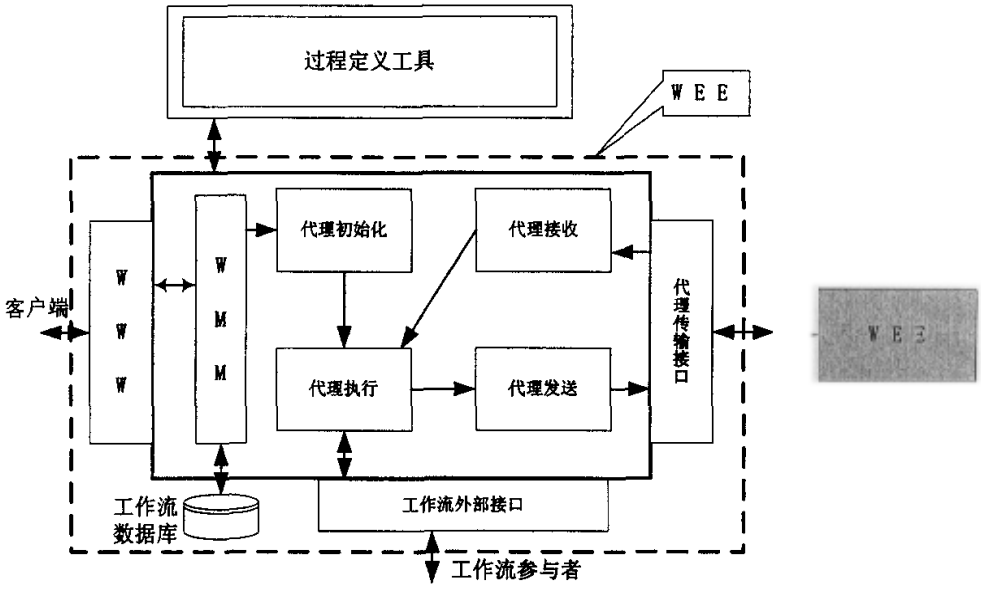

图 3-3　CWE 体系结构示意图

如图 3-3 所示，CWE 体系结构可以分为两部分，过程定义工具（Process Definition Tools，PDT）和工作流执行环境（Workflow Execution Environment，WEE）。而工作流执行环境又可进一步分为工作流管理环境（Workflow Management Environment，WME）和移动代理运行环境（Mobile Agent Runtime Environment，MARE）。

PDT 是一个工作流过程定义工具集。过程定义工具处理过程分析，过程建模和过程定义任务。在这个过程中，确定参与工作流过程的角色和过程对象，并建立一个原数据（meta-data）模型表示这些对象。

WME 包括一个工作流管理模块（Workflow Management Module，WMM）

和一个工作流数据库组成。WMM 模块的主要功能是接收来自过程定义工具的过程定义，并将它们保存到工作流数据库当中；同时，它还负责响应来自客户的请求部署相应的移动到代理运行时环境，启动一个工作流过程；另外，就是负责接收那些在本站点初始化的代理执行状态信息，以及监控代理的执行。

MARE 提供代理所需的移动、执行、生命周期管理和其它一些功能。它能够通过代理传输接口（Agent Transport Interface，ATI）发送和接收工作流代理。支持代理与外部工作流参与者之间的交互。这些参与者可能是人或其它应用工具。最重要的是，MARE 以虚拟机的方式提供了代理与系统平台无关性支持。这使得代理在异构网络环境中执行成为可能。

基于移动代理的 CWE 的体系结构，能够很好的适应基于 Internet 的电子商务应用需求。它具有以下优点：

1) CWE 是一种支持松散耦合的体系结构。松散耦合关系是基于互联网电子商务的主要特征。在传统工作流系统中，工作流角色与工作流参与者之间的映射关系以及工作流应用的部署都是在过程定义过程中完成的。因此，当参与工作流的任何一方发生变更的时候，就要从新进行过程定义。CWE 采用移动代理技术，可以将关系映射延迟到运行时。这样就便于当松散耦合关系变化时，动态的作出调整。这对于增加工作流系统的灵活性是很重要的。

2) 支持异构分布式平台。代理的迁移和执行是在代理运行时环境（MARE）中进行的，MARE 为代理提供了一个虚拟机器，使其不依赖于系统平台。每一个工作流参与者仅仅需要一个 MARE，而不是整个工作流管理系统。

3) 提供自治能力。一方面，自治能力作为移动代理的本质特性之一，使得工作流代理能够根据当前的环境和自身的状态决定下一步目标或操作。另一方面，在松散合作关系中的每个参与者通过改变本地数据和策略能够影响工作流的执行过程。

4) 支持并行计算。在 MARE 的支持下，代理可以复制几个自己并行的执行同一个任务，以缩短整个过程执行时间。这些复制代理，在结束它们的任务后能够与主代理合并。

5) 支持恢复与容错。工作流代理能够持久化保存在存储设备上，对它的恢复是

很简单的。这种机制增强了整个工作流系统的容错能力。

6) 支持异步执行。工作流代理能够在迁移到目标后，断开网络连接。在执行完任务后，再恢复连接。这对于构建在不稳定的互联网环境的应用来说有着重要的意义。

CWE 是一个基于代理的中间件体系结构，其多代理系统组成结构如图 3-4 所示。

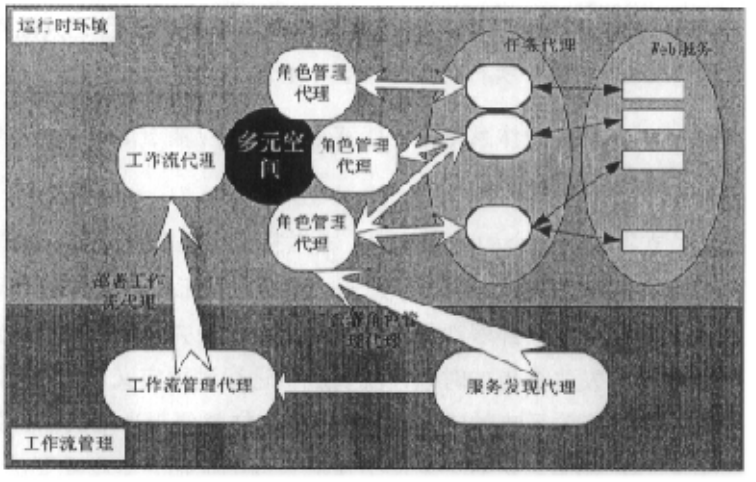

图 3-4　CWE 多代理系统组成示意图

运行时环境是工作流实例化和实际运行的空间。它由三种代理组成：工作流代理（Workflow Agent，WfA）、任务代理（Task Agent，TA）和角色管理代理（Role Management Agent，RMA）。CWE 正是通过这些代理之间的协同完成一个工作流。

工作流代理

系统为每个工作流实例创建一个工作流代理。它接收工作流管理代理传递过来的数据和过程定义信息，并代表相应的工作流实例完成过程定义工作内容。

任务代理

一个工作流过程由一到多个活动（activity）组成。在工作流运行时，系统

为过程实例的每个活动创建一个任务代理。任务代理必须确保活动所代表的任务被执行。

角色管理代理

角色管理代理通过执行一个或多个服务来担当工作流执行过程中的一个角色。有关该角色的定义在工作流策略文件中进行描述。

另外，在运行时环境中还有一个多元空间。它是工作流代理与角色管理代理进行协同工作的环境。它为多个代理进行协调提供了一系列基本服务，如事件注册，通讯等。多元空间是一个抽象概念，它是多代理进行协商交互的系统环境。在系统的不同实现中，多元空间也会有不同的表现形式。例如，在利用 Java 技术的实现中，一个多元空间可能就是一个 Java 空间（Java Space）。

在 CWE 体系结构中，并没有将 Web 服务封装在移动代理中，我们这样做主要是出于效率上的考虑。虽然代理本身拥有计算能力，但通常不会用它进行大量运算，而是调用本地服务对象来完成实质性的计算任务。由于要屏蔽异构平台的差异，代理语言一般采用解释执行的方式，效率不高。另外，高效的计算软件往往与系统紧密耦合。所以，我们采用 Web 服务调用的方式。这样，Web 服务就可以采用本地语言和环境来实现，在此之上提供一个代理调用的接口。

另一个重要原因就是重用遗留（legacy）软件。在进行信息化建设的历程中，我们已经积累了大量的软件系统，这是一笔巨大的财富资源。由于，它们使用是比较的陈旧的技术，通常没有考虑到与其它系统的互连和互操作。与它们的集成有一下三种途径：一是重写该软件，这显然代价太大；二是开发一个独立的转换软件，在代理通讯语言和这些遗留软件的本地协议间进行翻译转换；三是利用封装技术，在原有的软件上加上一块代码，使之可以与代理通讯。考虑到效率和编写转换器较为困难，我们认为第三种途径较为合适。

另外，这种调用机制也为与其它开放系统（如 CORBA）进行互操作流下了余地。

工作流管理模块包括工作流管理代理（Workflow Management Agent，WMA）和服务发现代理（Service Discovery Agent，SDA）。

工作流管理代理

负责接收工作流定义工具生成的工作流描述。并根据这个描述进行协同层的工作流代理的部署。它负责实例化一个工作流实例。通过创建一个工作流代理和传递所需的数据条目给这个代理。

服务发现代理

负责从现存的 Web 服务（包括本地和网络服务）抽取特征信息，并将这些信息保存在共享数据库中。另外，SDA 还负责部署角色管理代理。

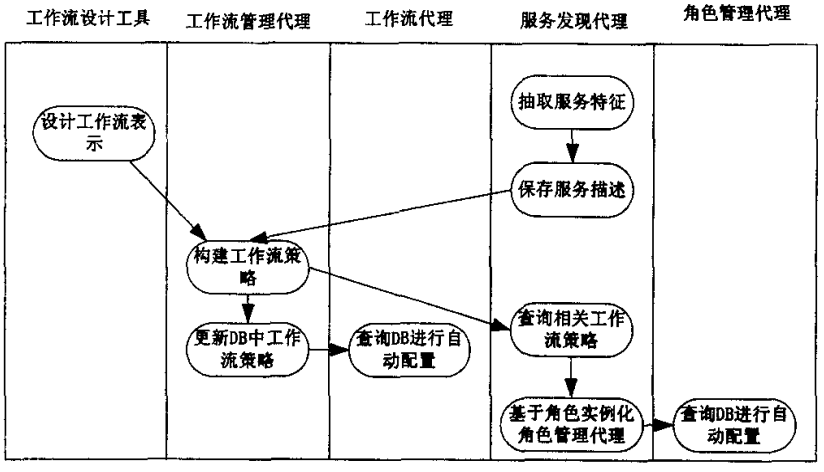

图 3-5　CWE 配置过程示意图

CWE 的配置过程，如图 3-5 所示，下面我们依据时序关系详细介绍整个配置过程。

1) 工作流管理代理接收工作流设计工具产生的工作流描述作为其输入。服务发现代理则负责在网上查找（如通过 UDDI 注册中心）适合的 Web 服务，并抽取其服务特征生成相应的服务描述作为工作流管理代理的输入。

2) 工作流管理代理根据接收的工作流描述和服务描述信息生成工作流策略描述。

3) 工作流代理查询数据库进行自我配置。

4) 服务发现代理查询工作流策略信息，基于工作流角色实例化角色管理代理。

5) 角色管理代理查询数据库进行自我配置。

从本质上讲，协同工作引擎是一个多代理系统，因此，我们还是从代理角度来论述协同工作引擎的工作机理。

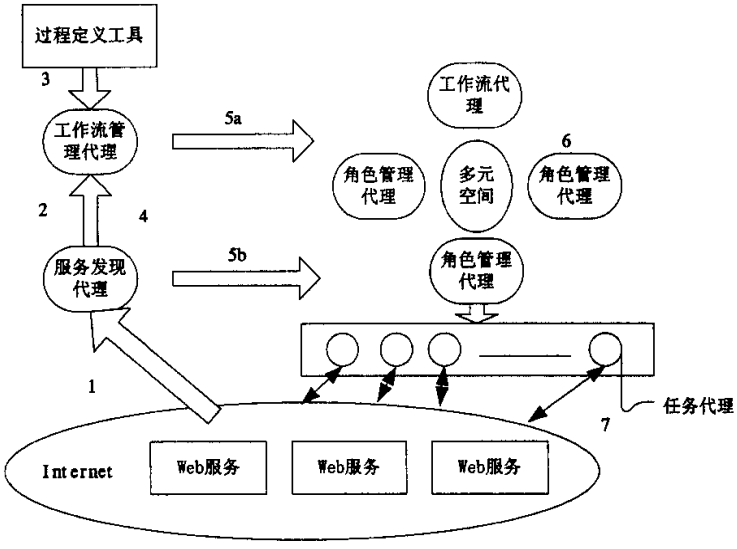

图 3-6　协同工作引擎工作原理图

如图 3-6 所示，协同工作引擎的操作过程共有 7 个步骤，分别用数字标出，下面我们详细介绍每个步骤：

第一部分，制定工作流策略，步骤 1~4：

1〉服务特征抽取。服务发现代理，利用服务访问协议（如：SOAP）访问服务描述，并抽取其中的服务特征。

2〉保存服务特征到数据库。

3〉过程定义。利用过程定义工具，生成一个计算机可以处理的过程描述。

4）生成工作流策略描述，并保存在数据库中。

这里需要说明的是，步骤 1 和步骤 3 没有必然的先后顺序的关系。服务特征抽取，是一个不定时长期的行为。网络上新的服务广告，以及原有服务的更新等，服务发现代理都可以通过基于语义 Web 服务的描述和 UDDI 注册中心的交互可以及时地发现，并通过步骤 2 将这些服务新保存在基于工作流的数据库中。在生成工作流策略描述的时候，可以应用这些服务描述。

第二部分，部署应用协同层，由步骤 5a 和 5b 组成。

5a）部署工作流代理。通过创建工作流代理实例，完成过程实例化。

5b）部署角色管理代理。创建角色管理代理，基于工作流策略描述，让它们在工作流扮演相应的角色。

第三部分，过程实例的执，由步骤 6、7 组成。

6）工作流代理与角色管理代理在多元空间的协同操作。

7）任务代理与 Web 服务的交互。任务代理与相应的 Web 服务绑定，并与其交互履行相应的服务职能。

3．4　流程语义描述

3．4．1 本体介绍

代理之间的有效的交互，依赖于对表达的相互理解，它又可分为两个方面：一是从一种表达语言翻译成另外一种表达语言；二是不同知识表达之间的语义共享。对代理交互来说，单纯的翻译是不够的，最重要的还是语义的共享。因为每一个知识库都具有对表达内容的意义的潜在假设。如果两个基于不同知识库的代理要进行交互，这些假设必须共享，以实现对知识表达的共同理解。本体就是用来进行知识表达技术，下面我们就来介绍有关本体的知识。

本体论（Ontology：o 大写）原是哲学的一个研究分支，研究客观事物存在的本质。它与认识论（Epistemology）相对，认识论研究人类知识的本质和来源。也就是说，本体论研究客观存在，认识论研究主观认知。而本体（ontology：o

小写）的含义是形成现象的根本实体(常与“现象”相对)。

在人工智能领域，知识建模必须在知识库和两个子系统之间建立联系：agent 行为（问题求解技能）和环境（问题存在的领域）。而长期以来，AI 的研究者较为注重前一个子系统，而领域知识的表达依赖于特定的任务。这样做的好处是只需要考虑相关的领域知识。但是，大规模的模型共享、系统集成、知识获取和重用依赖于领域的知识结构分析。因此，进入九十年代以来，与任务独立（task-independent）的知识库（本体）的价值被发现，并受到广泛关注。

随着本体研究的日趋成熟，关于本体的概念和术语用法也渐趋一致。下面我们首先给出一个具有代表性的本体定义：

本体是对领域知识“概念化”（Conceptualization）的明确表达，使用于描述和表达某一领域知识的一组概念或术语，它可以用来组织知识库较高层次的知识抽象，也可以用来描述特定领域的知识。

与本体相关的概念和术语：

本体论（Ontology，o 大写）：特指哲学的分支学科。

概念化（conceptualization）：指某一概念系统所蕴涵的语义结构，它是对某一事实结构的一组非正式的约束规则。它可以理解和/或表达为一组概念（如实体、属性、过程）及其定义和相互关系。

本体理论（ontological theory）：表达本体知识的逻辑理论，它是一种特殊的知识库，是本体知识所赖以存在的介质，强调的是具体的产品（designed artifact）。而“概念化”强调的是语义结构本身，是从具体的产品中抽象出来的对应的语义成分（semantical counterpart）。

本体约定（ontological commitment）：对使用某一本体所定义词汇并与其含义保持一致的承诺。

本体工程（ontological engineering）：知识工程的分支，它研究如何用本体论的原则来构造本体理论。

总的来说，构造本体的目的都是为了实现某种程度的知识共享和重用。下面我们将本体的作用总结如下：

- 本体的分析澄清了领域知识的结构，从而为知识表示打好基础。本体可以重用，从而避免重复的领域知识分析。
- 统一的术语和概念使知识共享成为可能。
- 通讯：主要为人与人之间或组织与组织之间的通讯提供共同的词汇。
- 互操作：在不同的建模方法、范式、语言和软件工具之间进行翻译和映射，以实现不同系统之间的互操作和集成。
- 基于本体分析能够为系统工程提供以下方面的好处：
  - ◆ 重用（re-usability）：本体是领域内重要实体、属性、过程及其相互关系形式化描述的基础。这种形式化描述可成为软件系统中可重用和共享的组件（component）。
  - ◆ 知识获取（knowledge acquisition）：当构造基于知识的系统时，用已有的本体作为起点和基础来指导知识的获取，可以提高其速度和可靠性。
  - ◆ 可靠性（reliability）：形式化的表达使得自动的一致性检查成为可能，从而提高了软件的可靠性。
  - ◆ 规范描述（specification）：本体分析有助于确定 IT 系统（如知识库）的需求和规范。

本体的分类方法很多，目前还没有能够被广泛接受的分类标准。如可以根据本体不同方面的属性（如形式化程度、目的和描述对象），可以对本体进行不同的分类。

如根据本体的形式化程度不同，可以把本体分为高度非形式化的（highly informal）、结构非形式化的（structured-informal）、半形式化的（semi-formal）和严格形式化的（rigorously formal）；根据本体的描述对象不同，可以把本体分为特殊领域本体（如医药、地理、金融等）、一般世界知识本体、问题求解本体和知识表示语言本体等。

但以下几个概念的定义意义明确，并从某种程度上提供了本体的分类方法：

- 领域本体（DOMAIN ONTOLOGY）：以某一领域为描述对象的本体（区别

于领域的问题和任务）。

- 问题求解模型（PROBLEM SOLVING MODEL）：以问题求解方法为描述对象的本体。
- 表示本体（REPRESENTATION ONTOLOGY）：以知识表示语言为描述对象的本体。在表示本体中，类、对象、关系、属性、槽等术语经过严谨的分析和定义。

从描述对象的类型来说，本体既可以用来描述简单的事实，又可以用来描述信念、假设、预测等抽象的概念；既可以描述静态的实体，又可以描述与时间推移相关的概念，如事件、活动、过程等。

从描述对象的范围来说，本体可以定义通用的、适合所有领域知识表示的术语，如空间、时间、部分等；也可以定义特定领域知识才使用的术语，如故障、肝炎等。

不同本体之间存在着差别，但它们在较高的抽象层次上（upper ontology）具有一些共同的特征：

- 世界存在着对象（object）；
- 对象具有属性（property or attribute），属性可以赋值（value）；
- 对象之间存在着不同的关系（relation）；
- 属性和关系随着时间（time）的推移而改变；
- 不同的时刻（time instant）会有事件（event）发生；
- 在一定的时间段上存在着过程（process），对象参与到过程当中；
- 世界和对象具有不同的状态（state）；
- 事件能导致（cause）其他事件发生或状态改变，即产生影响（effect）；
- 对象可以分解成部分（part）。

从客观的意义上说，本体的描述是和特定的任务和目的无关的，但是我们使用本体总是有一定的任务背景，因此对于所描述的知识的选择是和特定任务相

关的。这也是不同顶层本体（upper ontology 或 top-level ontology）之间存在差异的主要原因。

面向对象的软件设计也依赖于适当的领域本体，对象、属性、方法或多或少会反映领域中与应用相关的方面，OO 系统中对于领域的分析往往可以在不同的应用程序中重用。OO 系统和本体强调了不同的侧面，但随着时间推移将逐渐融合。由于信息系统要对广泛的领域进行建模，领域本体将在软件系统中起着与 AI 领域中类似的重要作用。

3．4．2 一个基于工作流的本体

本小节我们将提出一个基于工作流的本体，以及采用面向对象分析和设计方法来构造基于工作流的本体。

3．4．2．1 基于工作流的本体

在工作流管理这样一个领域上下文中，代理之间的通讯问题与工作流管理的相关概念有着密切联系。事实上，我们设计 CWE 系统中代理之间的通讯协议正是基于一个基于工作流的本体构造的。

工作流本体是从工作流管理系统有关概念及其相互关系抽象而来。下面我们根据 WfMC 定义，简要工作流管理系统中基本概念及其相互关系。

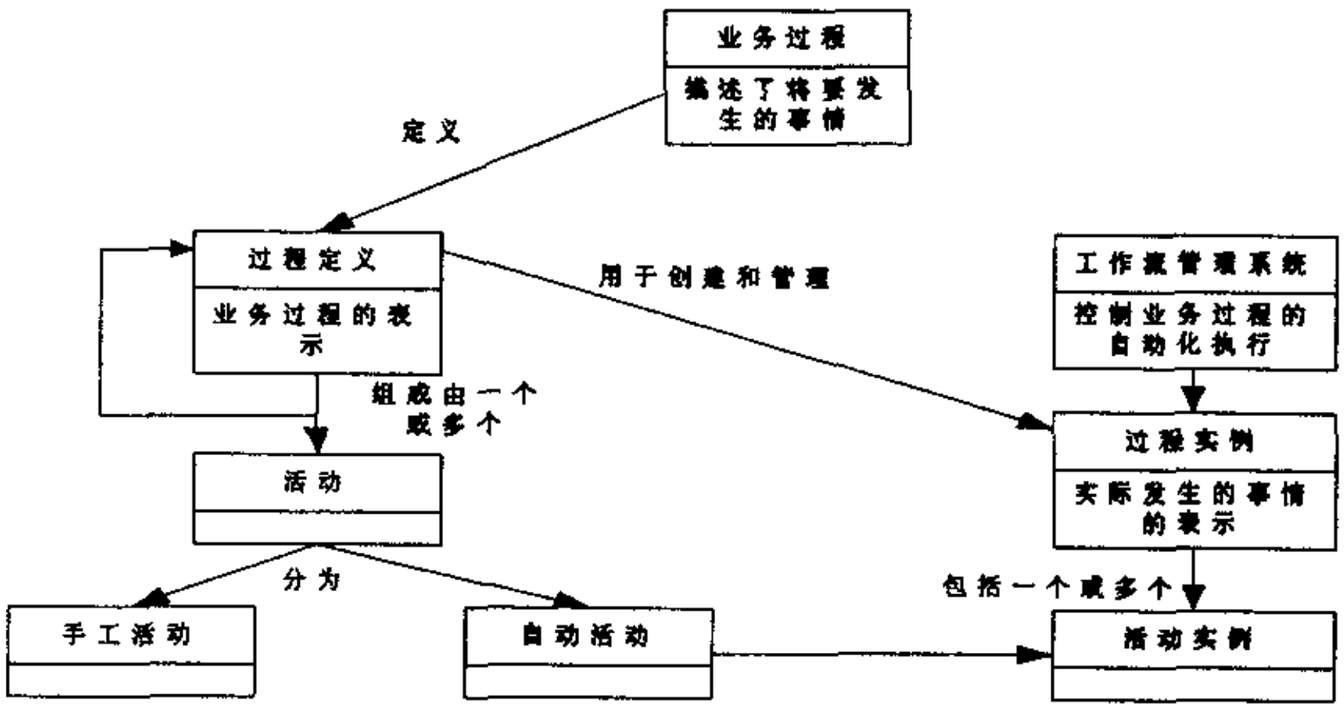

图 3-7 工作流基本概念及其相互关系

如图 3-7 所示，在工作流管理系统中，业务过程描述了在一个业务流程中将要发生事情。过程定义是用形式化方法对业务过程描述，以实现将业务过程全部或部分的自动化。一个过程由一个或多个活动组成，这是一个递归定义，即一个活动可以是一个子过程。活动描述了业务过程中一个逻辑步骤内将要完成的工作内容。活动分为手工过程，即需要人或软件代理参与完成的活动，和自动过程（通过调用相应的应用程序自动完成）。工作流管理系统是控制工作过程实例执行的软件系统。过程实例是工作流管理系统根据过程定义创建的，它表示过程在一次实际执行期间所发生的事情。一个过程实例由一个或多个活动实例组成。一个活动实例代表一个运行期间的活动。

针对工作流的基本概念及其相互关系，我们给出一下基于工作流的本体示意图。其中使用的概念是基于我们在第一章中给出工作流基本术语和概念。

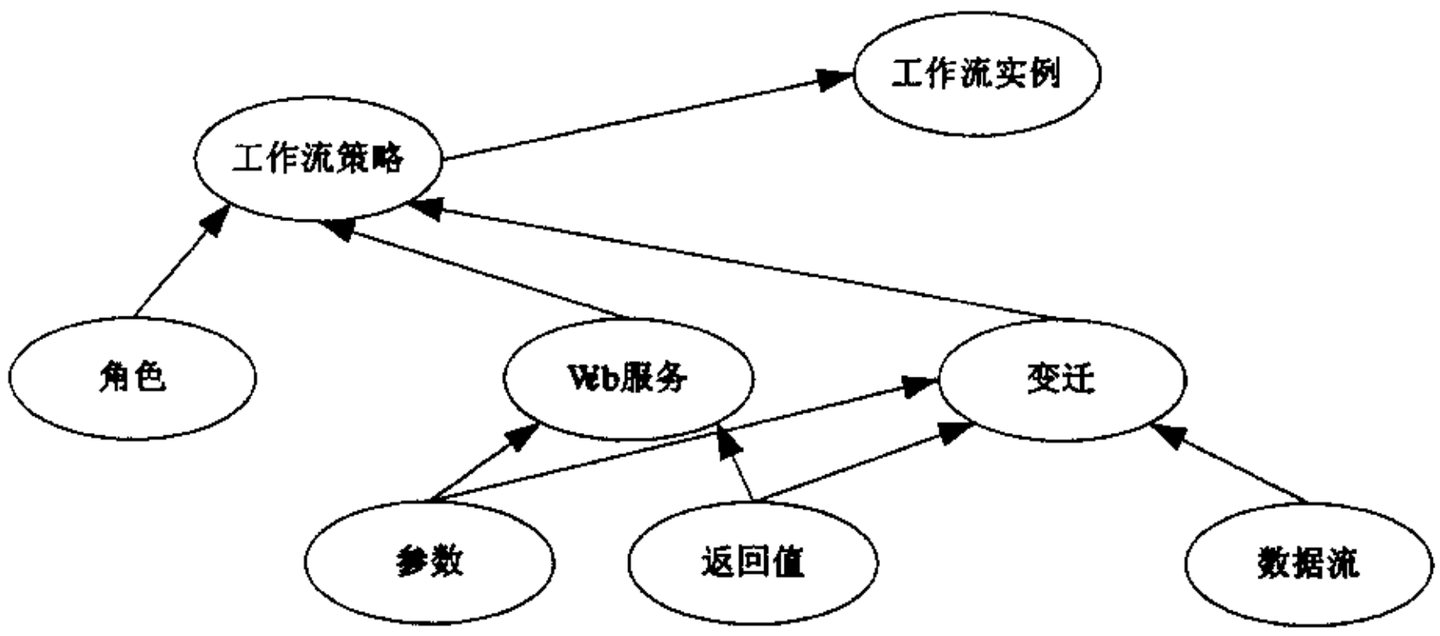

图 3-8　基于工作流的本体示意图

3．4．2．2 面向对象的本体设计

上一小节中，我们通过抽象工作流管理系统的基本概念及其相互关系给出了一个基于工作流的本体的基本结构。正象大多数的本体都是可扩展的那样，我们给出的这个本体也是可扩展的。但是这个本体是如何得出的呢？本文中，我们采用面向对象的分析和设计技术，对领域知识进行建模得出本体结构，对这个本体的扩展也可以采用这种方法。下面我们还是首先来简单回顾一下有关面向对象的理论和技术基础知识。

面向对象的思想认为客观世界的问题都是由客观世界的实体及其相互关系构成。这与本体抽象表达领域知识的思想不谋而合。在面向对象的方法中将客观世界的实体称为“对象”。显然，对象因问题域的不同是不固定的，一本书可以是一个对象，一个图书馆也可以是一个对象。在面向对象方法中，有三个核心概念：封装、继承和多态。

封装将客观世界的信息和处理这些信息的功能封装在一起，形成一个能动的实体，称为对象。对对象的使用和访问不必知道对象内部实现的细节，而只需要该对象提供的外部访问接口。

继承将一般和特殊的关系模型化。例如，一个过程定义和过程实例就是一般和特殊的关系，对与这种关系我们就可以用继承的概念进行表示。

多态是指特定功能的多种实现方法。

基于工作流本体是代理之间的共享知识库。事实上，代理之间的通信就是基于由工作流策略、角色、服务和数据流组成的这样一个领域知识上的行为推理。

下面我们使用 UML 类图重新给出本体的静态视图，如图 3-9 所示。

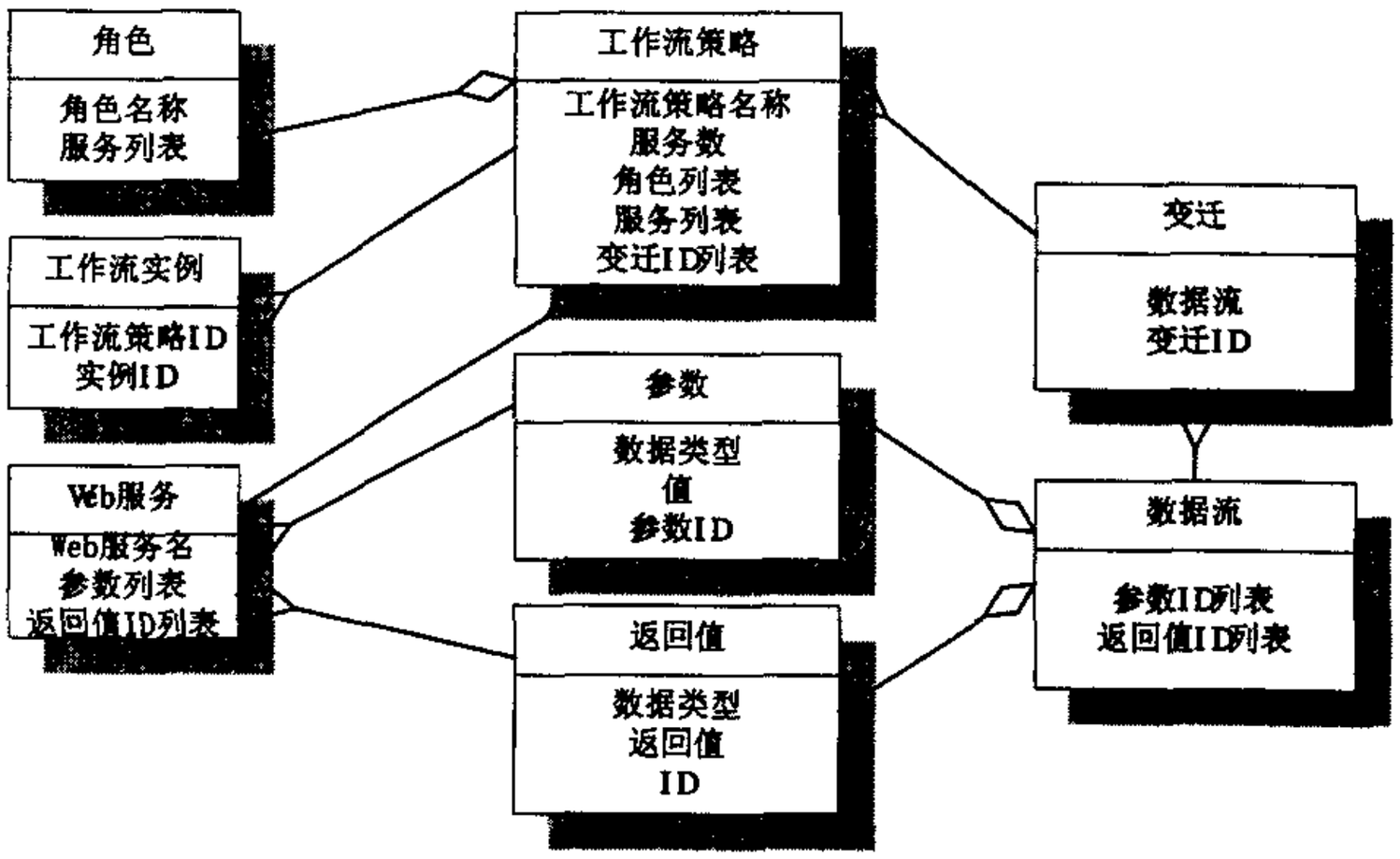

图 3-9　面向对象本体示意图

工作流策略是本体的核心。代理（基于组件的 Web 服务）首先需要知道工

作流策略。在工作流策略中的每一步关联着一个角色和一个指定的 Web 服务实现。每个服务有一个或多个参数（前置条件）或返回值（后置条件）。工作流策略可以进一步定义为一个由参数和返回值的子集，它们构成了在每一步中间流转的数据流。之所以定义数据流是因为一个服务可能返回比下个服务所要求的更多的信息内容。因此，可以把这些需要的信息（返回值）的子集定义为数据流传给下个服务。另外，多个并行的服务可能拥有同一个后趋服务。在这种情况下，需要合并多个服务的返回值。

3.5　在线服务的工作流操作

我们在上一章，论述了协同工作引擎的三个核心技术，其中 Web 服务技术是作为应用形式，也就是说协同工作引擎的设计是为我们提供一个操作 Web 服务的工作流框架。那么，对这些 Web 服务的操作有哪些呢？它们都发生在什么时候，以及这些操作的顺序是怎样的呢？

当一个任务初始化或者服务结束事件被捕获的时候，以及一个工作流没有发生任何错误正常执行结束时，会进行对代理之间以及代理对 Web 服务的标准操作。下面我们用一个 UML 的序列图描述这些操作。

如图 3-10 所示，初始化一个任务或一个服务的结束事件发起一个工作流实例。工作流代理侦听器捕获这一事件，基于它的工作流策略创建一个工作流实例唯一标识 ID。从这一时刻开始，角色管理代理将参与协同完成整个工作流。角色管理代理执行相应的活动，在它们运行结束后，角色管理代理向事件服务器发送结束事件。这一事件将触发下一个（或多个）服务的执行。

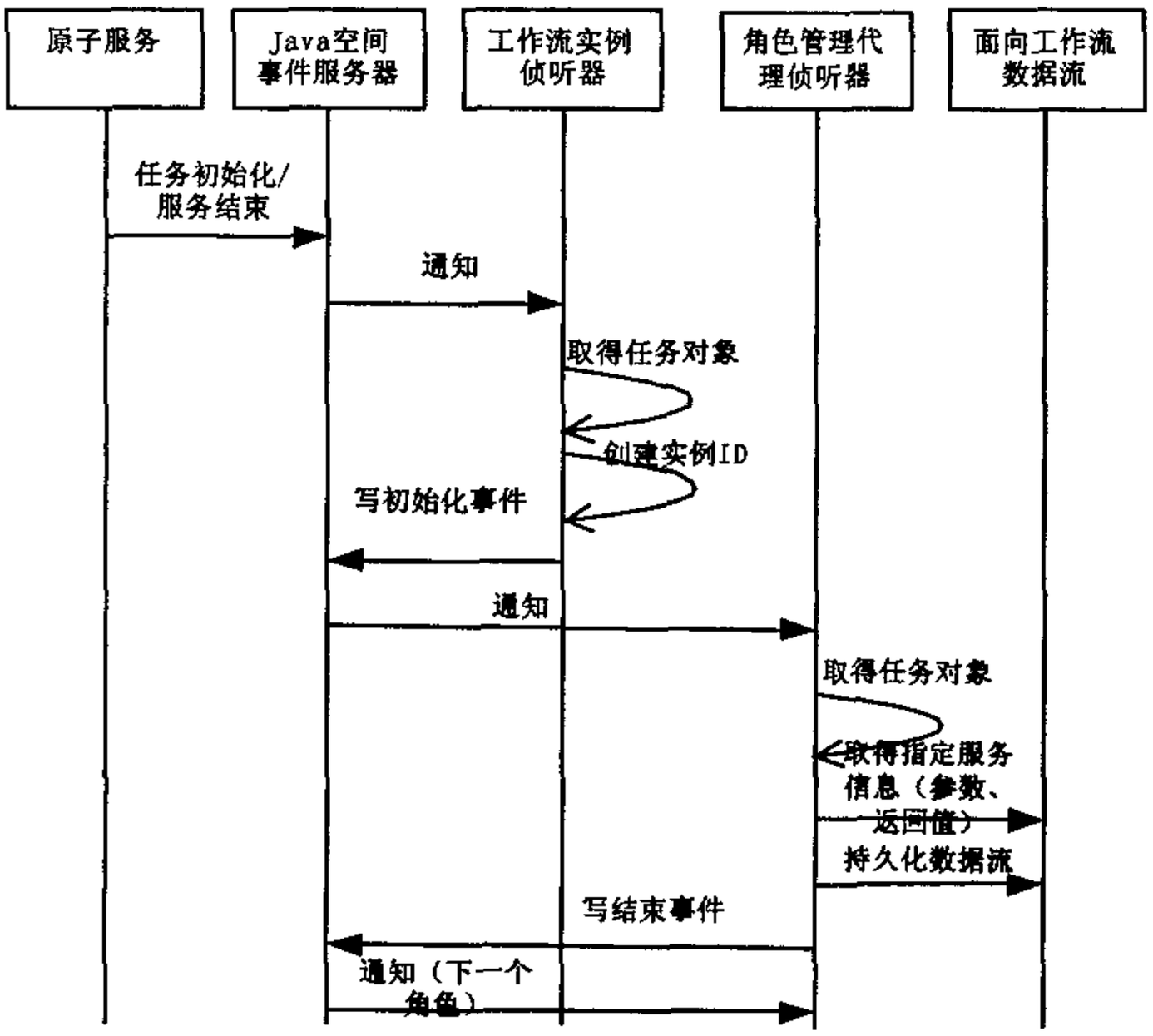

图 3-10　工作流操作示意图

3．6　本章小结

本章分析传统工作流管理系统存在局限性及其问题的根源基础上，论述了基于移动代理的解决方案的优点及其构建方法。基于以上的分析和讨论，提出了一个基于移动代理的工作流管理系统体系结构——协同工作引擎体系结构，及其多代理系统组成，并详细论述了其工作原理。

其次，基于代理通讯语义信息描述的需求，引出了本体理论和技术，通过对工作流领域概念的分析，提出了一个基于工作流的本体，并以面向对象的分析与设计方法，重构了基于工作流的本体。

最后，利用 UML 的序列图描述了基于工作流对在线 Web 服务操作过程。

第四章 工作流建模

在本章中，我们首先介绍描述工作流各个方面视图模型，包括三个静态模型和两个动态模型。然后，我们将详细说明如何将这些视图映射到 XML 空间，并最终转换成工作流数据库的关系模型。

4．1 基于 UML 的工作流建模

在第一章绪论我们介绍了几种工作流的建模方法。然而这些模型要么过于复杂难于实现，要么不适合于基于移动代理的工作流系统结构。下面我们提出的这种工作流建模方法是基于软件工程中的统一建模语言（UML）进行过程定义描述的。UML 采用面向对象的思想，对现实世界进行描述是一种很直观的方法，而且它有标准的建模工具，我们可以很容易的对工作流过程建模，并且将它转化为代理可以理解的格式。

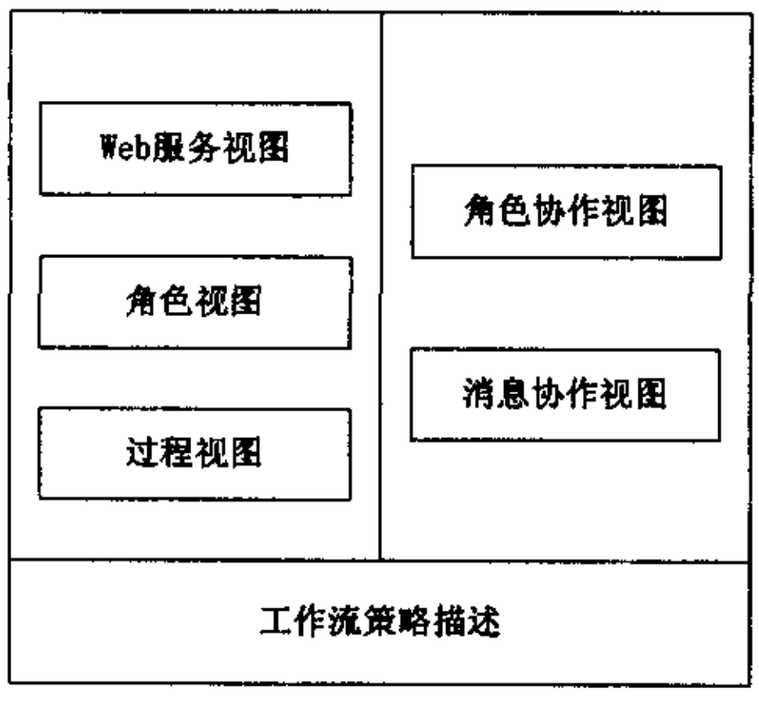

图 4-1　基于 UML 的工作流描述模型

用 UML 描述工作流过程分为三个部分：

1)　静态视图。包括 Web 服务视图，角色视图和过程视图。服务视图描述了一个业务过程中需要的 Web 服务及其相互关系。其中每个 Web 服务完成一定的业务功能。角色是个任务集合的抽象，每个角色代表一个工作流活动执行者所要完成的任务。在协同工作引擎中，任务是由任务代理调用执行相应的 Web 服务来完成的。角色视图描述了角色与 Web 服务之间的关系。

2) 动态视图包括角色协同视图和消息协同视图。。角色协同视图，给出了参与工作流执行的各个角色在整个过程实例运行期间，运行的时序关系以及在各步骤中完成的任务。事实上，在每个任务之间两个或多个协同的角色都要相互传递一定的信息，以告知各个任务执行的状态和结果，这就是在角色之间传递的数据流（Data Flow）。消息协同视图在角色协同视图的基础上，给出了在工作流执行期间，在角色管理代理之间进行的消息交换。

3) 工作流策略描述了过程定义的静态信息如工作流名称，工作流类型等信息，以及运行时角色协同关系，任务调度时序以及数据流等信息。

4．2 静态视图建模

为了便于以下展开论述，我们用一个实例来说明工作流建模方法。我们考虑这样一个场景，用户通过互联网订购股票，一个中介机构代理用户管理投资帐户信息，并接受用户委托完成股票交易。

4．2．1Web 服务视图

为了完成网上订购股票交易，我们需要提供如图 4-2 所示的 Web 服务。

图 4-2　Web 服务视图

getTradeRequest()：取得交易请求信息。

getUserInfo()：取得进行交易的用户信息。

cancel()：取消交易。

searchPortfolio()：查询投资帐户信息。

updatePortfolio()：更新投资帐户信息。

makeTrade()：执行交易。

4．2．2 角色视图

为了完成整个交易过程需要 3 个角色：

- 客户接口角色

它是用户与计算机系统的接口。通过该接口用户可以取得交易信息、用户信息和取消交易等服务。

- 投资管理角色

投资管理角色顾名思义负责客户投资信息的管理和维护工作。

- 交易角色

负责执行用户交易指令，完成交易活动，并更新投资信息。

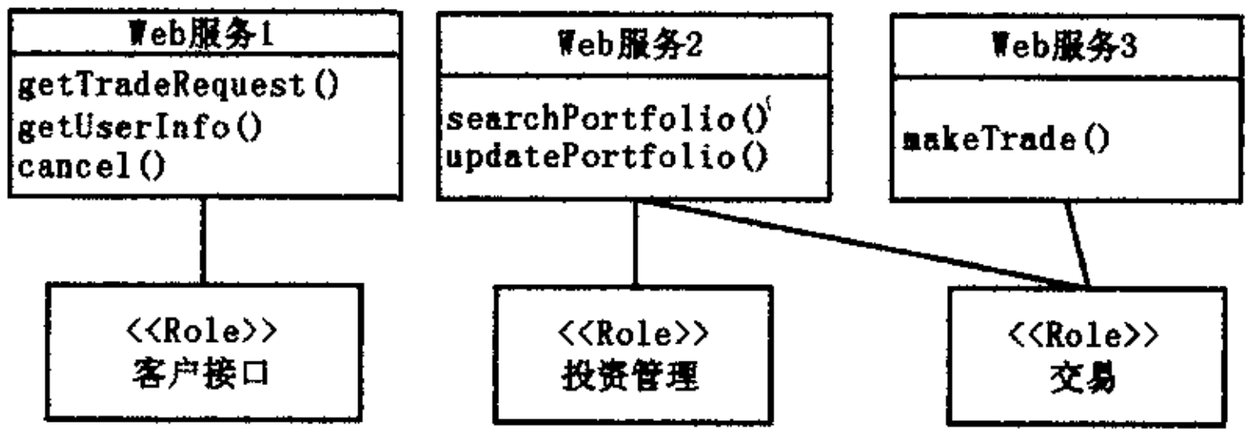

图 4-3　角色视图

针对这个三个角色，在工作流过程运行期，服务发现代理部署（并创建）相应的角色管理代理，客户接口角色管理代理、投资管理角色管理代理和交易角色管理代理。

4．2．3 工作流过程结构视图

在线股票订购业务过程结构视图如图 4-4 所示。

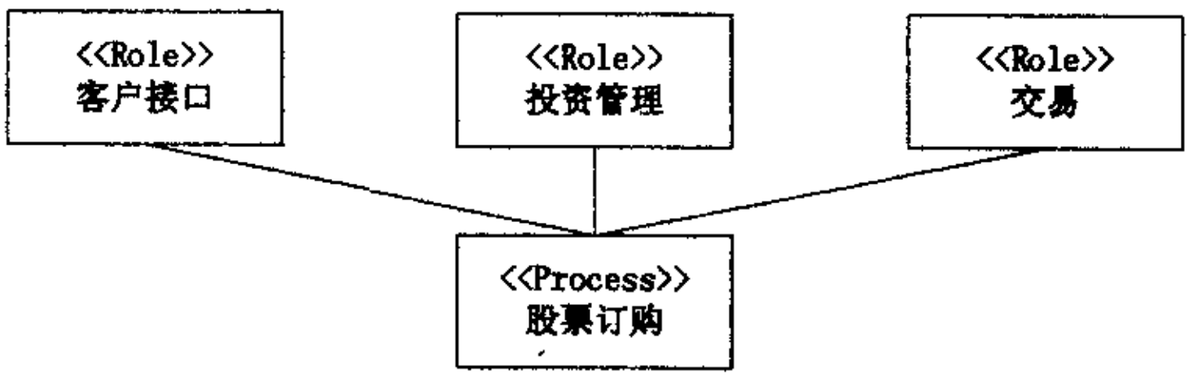

图 4-4　过程结构视图

通过该视图，我们可以知道一个业务过程涉及哪些角色，股票订购过程是

由客户接口、投资管理和交易三个角色之间通过相互协调关系构成的。而通过图 4-3 的角色视图，我们可以知道每个角色提供哪些服务（完成哪些任务），再由图 4-2 的服务视图可以进一步确定每个服务完成整个业务过程的哪些功能。这样，通过三个不同侧面的静态视图结构，构造了整个工作流过程静态结构。

4．3 工作流动态视图建模

静态结构视图，描述了业务过程各个组成要素的关系及其构成。但仅仅有这些静态视图，我们还不能确定工作流执行过程。下面，我们再给出两个动态视图，角色协同视图和消息协作视图，来进一步刻画工作流运行时期的特性。

4．3．1 角色协同视图

角色静态视图，仅仅从服务内容构成方面对角色进行描述。而角色协同视图，则描述了过程运行时，角色之间的协同关系。

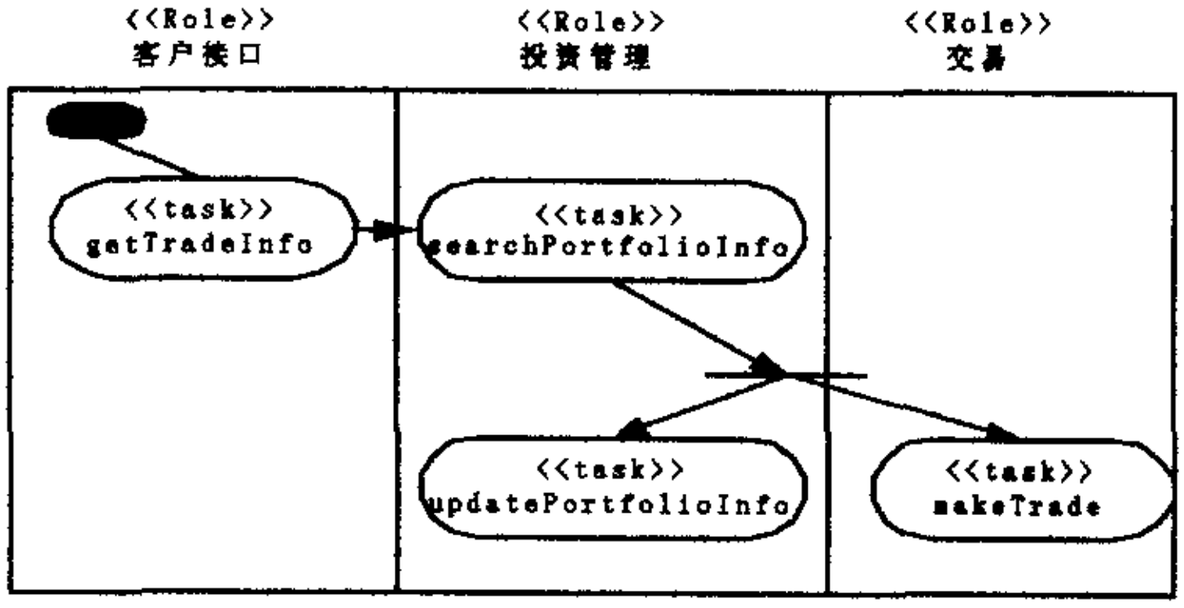

图 4-5　角色协同视图

如图 4-5 所示，过程实例运行期间，各角色管理代理相互协调完成工作流的过程如下：

1)　过程实例完成初始化任务后，客户接口管理代理创建任务代理调用 Web 服务 1 完成取得交易信息（getTadeInfo）的任务。

2)　投资管理角色管理代理捕获（Web 服务 1）服务结束的事件，调用 Web 服务 2 完成查询投资账户信息（searchPortfolioInfo）的任务。

3) 投资管理角色管理代理的调用 Web 服务 2 更新用户投资账户信息（updatePortfolioInfo），同步地交易角色管理代理调用 Web 服务 3 完成在线订购股票交易（makeTrade）。

4．3．2 消息协同视图

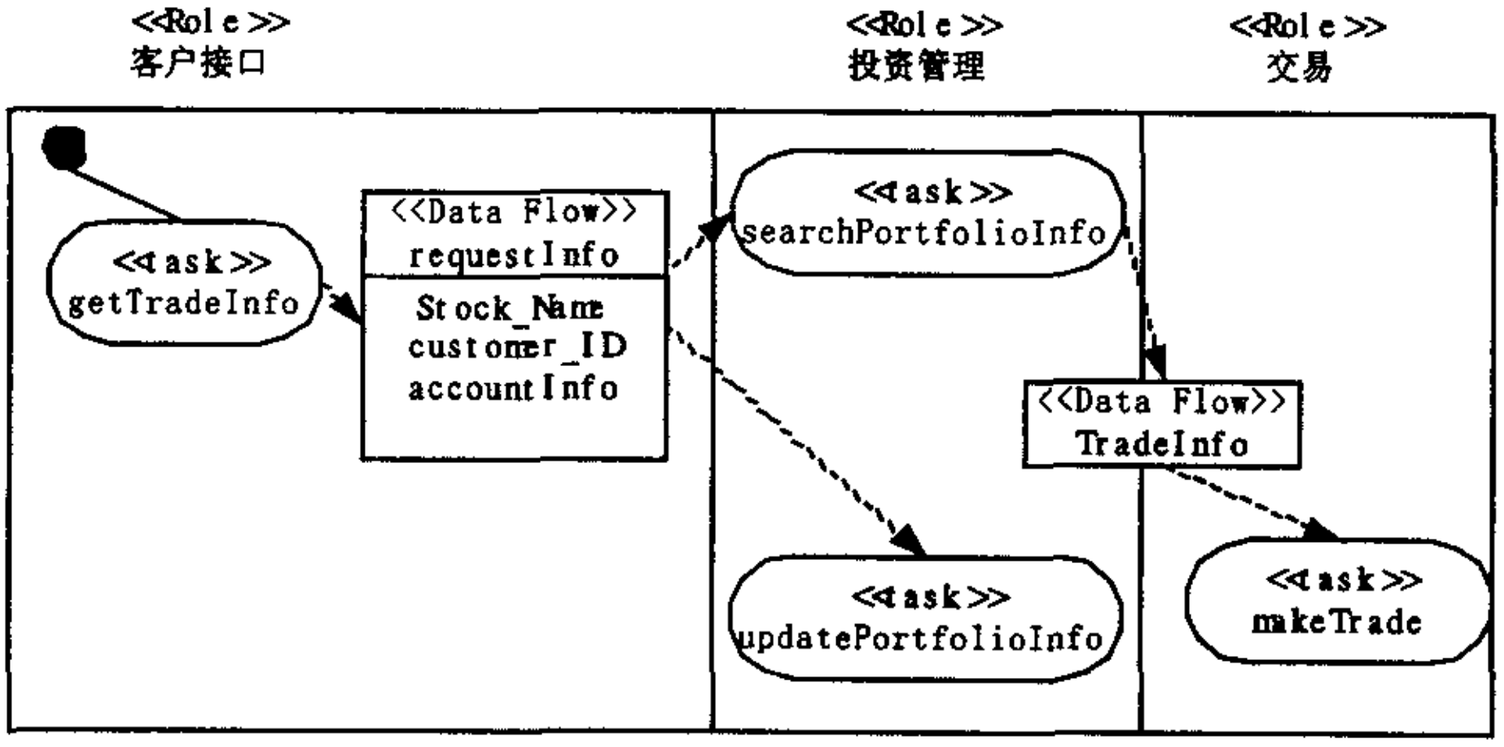

图 4-6　消息协同视图

如图 4-6 所示，在获取交易信息任务（getTradeInfo）结束后，在客户接口角色和投资管理角色之间传递的消息为数据流 requestInfo，它包含了股票名称、客户 ID 和账户信息等。它将作为查询投资账户信息任务（searchPortfolioInfo）和更新投资账户信息任务（updatePortfolioInfo）的输入。查询投资账户信息任务（searchPortfolioInfo）根据客户信息查询相应的投资账户信息。查询投资账户信息任务（searchPortfolioInfo）执行结束后，将数据流 TradeInfo 传给交易角色执行股票交易任务（makeTrade）。

两个动态视图分别从过程实例运行期间，角色之间协作关系和消息交换两个角度刻画了整个过程执行的动态过程。我们可以据此将业务逻辑分别封装在不同的 Web 服务当中，通过角色协同视图和消息协同视图构建工作流策略，并以此控制过程实例的运行。

4．4 构建基于工作流数据库模型

基于工作流的共享数据库（Share Database）是支撑协同工作引擎运行的底

层数据基础。共享数据库中保存了有关工作流策略相关信息，以及工作流实例运行时的各种状态、数据和事件信息。这个数据库同时也体现了基于工作流本体中的对象关系。协同工作引擎的数据库模型设计中实体关系如图 4-7 所示。

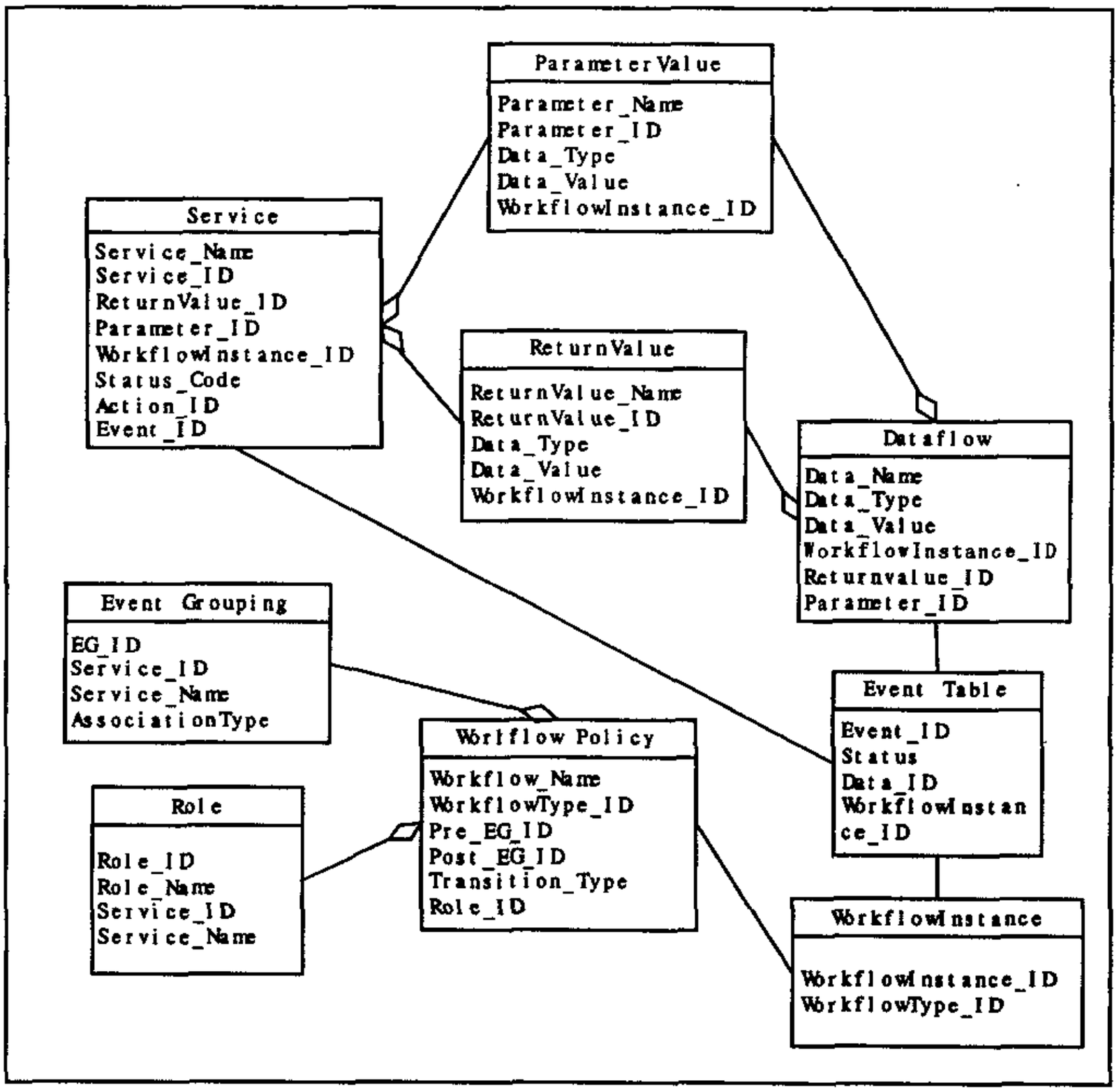

图 4-7　基于工作流的数据库模型

基于工作流的数据库模型，包括三个部分：工作流策略、服务和运行时数据。

- 工作流策略

这部分信息主要由三个表组成：工作流策略表、事件组表和角色表。

工作流策略信息包含工作流名称、工作流类型、前置事件组和后置事件组、变迁类型和角色等。流程名称和流程类型描述了工作流的基本属性信息。前置事件组和后置事件组，则表明了某个变迁前后发生的事件组，这里事件实际上

是某个 Web 服务的调用。数据流则指定了两个服务之间变迁过程中传递的数据。事件组是由一个或多个事件组成的，每个事件与某个 Web 服务相关联，同时事件组中还定义了这些事件之间的逻辑关系（如 AND，OR 等）。角色也与特定的服务相关联，一个角色可以同时关联到多个 Web 服务上。

- Web 服务

Web 服务信息，包括 Web 服务表、参数表和返回值表。同时，参数表和返回值表也是构成运行时数据，数据流的重要内容。

Web 服务信息包括，Web 服务基本属性信息，服务 ID、服务名称、参数和返回值等，以及 Web 服务所关联的工作流过程实例、活动和事件等。参数信息，包括参数名称、ID、数据类型和参数值，以及关联的过程实例等。返回值，包含的基本信息同参数。

- 运行时数据

运行时数据，是构成工作流实例运行态的动态数据，包括数据流、事件表和工作流实例表。

几乎在每次变迁发生的时候，在调用者与被调用者之间都要传递一些数据信息，我们把这些数据称为数据流，并保存在数据流表中。构成数据流信息的还有两个表，参数表和返回值表。每个参数或返回值，由实例运行过程中动态产生并保存到相应的表中。

事件表保存了所有从实例初始化到结束过程中所发生的事件，其中全局事件由工作流代理负责注册（保存），而每个任务（Web 服务）的事件由相应的角色管理代理负责。

工作流实例表保存了每个工作流实例 ID 号。每个工作流实例与特定的资源绑定在一起。每个工作流可以有多个工作流实例在同时运行。因此，通过工作流实例表，建立工作流 ID 和工作流实例 ID 的对应关系，可以知道每个工作流实例是为哪个过程定义创建的，并通过 ID 区分每个实例。

4. 5 基于 XML 的工作流策略描述

工作流策略信息包含，如流程定义信息、前置/后置事件组、数据流等信息。

在线股票订购工作流，过程定义（部分） XML 文件格式如下：

```
<ProcessDefinition>
<WorkflowInformation>
    <WorkflowName> StockOrder </WorkflowName>
    <WorkflowTypeID> On-line Stock .. </WorkflowTypeID>
    <Pre_EventGrouping_ID> UniqueID </Pre_EventGrouping_ID>
    <Post_EventGrouping_ID> UniqueID </Post_EventGrouping_ID>
    <Data_ID> UniqueID </Data_ID>
    <Transition_Type> SINGLE </Transition_Type>
    <Role_Name> Customer </Role_Name>
</WorkflowInformation>
<Pre_EventGrouping>
    <AssociationType> AND </AssociationType>
    <Event>
        <ServiceName> getUserInfo </ServiceName>
        <ServiceID> UniqueID </ServiceID>
    </Event>
</Pre_EventGrouping>
<Post_EventGrouping>
    <AssociationType> AND (Transition) </AssociationType>
```

```
<Event>
    <ServiceName> verifyUser </ServiceName>
    <ServiceID> UniqueID </ServiceID>
</Event>
</Post_EventGrouping>
<DataFlow>
    <DataName> UserInfo </DataName>
    <DataType> STRING </DataType>
    <ReturnValue_ID> UniqueID </ReturnValue_ID>
    <Parameter_ID> UniqueID </Parameter_ID>
</DataFlow>
</ProcessDefinition>
```

4．6 本章小结

本章详细描述了利用 UML 工具进行工作流建模的过程和方法。3 个静态视图，描述了工作过程流过程静态结构。2 个动态视图，则描述了过程执行的动态过程。而数据库模型，则给出了一个基于工作流的数据中实体及其相互关系。最后，以实例形式给出了一个完整的工作流策略 XML 文档。

第五章 协同工作引擎的原型设计

为了验证 CWE 体系结构的可行性，我们设计了一个原型系统。该原型系统是基于 IBM 的 Aglets 系统作为开发和代理运行平台。为简化起见，我们只构建了运行时期工作流系统。

5．1 开发环境

操作系统：Windows 2000

JDK：SUN JDK1.3.1

Aglets：IBM Aglets2.0.2

- JDK 的安装和配置

  假设 JDK 的安装目录为 C:\jdk1.3.1，则相关的环境变量配置如下：

  SET JDK_HOME=C:\jdk1.3.1

  SET JAVA_HOME=C:\jdk1.3.1

  SET CLASSPATH=.;%JDK_HOME%\bin

- Aglets2.0.2 的安装和配置

解压缩 Aglets-2.0.2.rar 包到目录 C:\agelets-2.0.2，运行 ant.bat（注意，你必须首先配置 JAVA_HOME 的环境变量。），这时在当前目录下会增加几个文件，运行 agletsd.bat(有时会提示找不到.keystore 文件，这时可将 bin\.keystore 拷贝到相应的目录下即可)。相关配置如下：

SET AGLET_HOME=C:\agelts-2.0.2

5．2 代理设计

5．2．1 工作流代理设计

工作流管理代理创建一个工作流代理启动一个工作流实例。在整个工作流

执行期间，工作流代理负责工作流全局状态和事件的管理和维护。以下是工作流代理的类设计原型：

```
public class WorkflowAgent extends Aglet {

    // 变量
    String Instance_id;
    String Workflow_Type;
    String Workflow_Name;
    // 构造方法，完成工作流代理的初始化工作
    public WorkflowAgent() { }
    // 方法，完成与角色管理代理的交互，以及工作流全局事件的监控
    public void handleMessage(Message message) {}//消息处理
    public void sentMessage(String string) { }//消息发送
    public void updateWInstanceState(String state){}//更新过程实例状态
    public void registEvent(String EventType){}//注册全局流程事件
}
```

工作流代理中，提供了事件注册方法用于对流程启动等全局事件进行注册。工作流代理并不对工作流执行进行控制，因此它主要负责那些带来非功能性改变的事件进行管理。

5．2．2 角色管理代理

角色管理代理代表一定的工作流角色参与整个工作流的协同过程。在一个工作流实例运行期间，它通过动态集成 Web 服务完成一个或多个任务。组成整个工作流的每个任务实际上是由任务代理调用相应的 Web 服务完成的。一个角色管理代理的类框架如下：

```
public class RoleManagerAgent extends Aglet {

    // 变量

    String Role_Id;

    String Role_Name;

    RoleManagerAgent roleManagerAgent;

    String Instance_Id;

    // 构造方法，完成角色管理代理的初始化工作

    public RoleManagerAgent() { }

    // 方法，完成与角色管理代理的交互，以及工作流全局事件的监控

    public void handleMessage(Message message) { }

    public void setMessage(String string) { }

    public void createTaskAgent(){}//创建任务代理

    public void registEvent(String Event_Type){}//注册流程步骤级事件

}
```

同工作流代理一样，角色管理代理中也提供了一个事件注册方法，用于对步骤级事件进行注册。

5．2．3 任务代理

任务代理代表过程实例中的每一个原子任务，它通过调用相应的 Web 服务完成其业务功能。

```
public class taskAgent extends Aglet {

    // 变量

    String Instance_Id;
```

```
    String Service_ID;
    String Service_Name;
    // 构造方法，完成任务代理的初始化工作
    public taskAgent() { }
    // 调用相应的 Web 服务
    public void runService(String Service_ID,String Dataflow_ID){};
}
```

5.3 实例运行

我们设计了如下实例：查找网络上电视机供应商提供的最优性价比的电视机。源主机通过本地工作流管理代理 Master 发出一个移动工作流代理 Slave，电视机供应商 Supplier 的地址由源主机指定。Supplier 通过提供一个本体角色管理代理 Service，以一个工作流角色的身份参与到工作流当中。

Supplier 的数据库提供的电视及信息，如表 5-1 所示。

表 5-1　Supplier 提供的电视机信息结构

字段	类型	说明
ID	长整型	商品编号
Mark	字符型	电视机品牌
Color	布尔型	是否彩色
Prize	长整型	价格
Size	整型	尺寸

启动 Aglets 服务器 atp://qiyue:4434，atp://qiyue:6666,atp://7777,atp://8888 分别作为源主机 A 和电视机供应商 B、C、D。模拟 Internet 环境。这里主要出于简化实验的目标。各供应商提供的电视机信息如表 5-2 所示。

表 5-2　各供应商提供的电视机信息

Mark	Color	Size	Prize(B、C、D)
ChangHon	True	21	1500,1600,1700
ChangHon	True	25	2000,2100,2200
ChangHon	True	34	3000,3100,3200
Haier	True	21	1600,1700,1800
Haier	True	25	2100,2200,2300
Haier	True	34	3100,3200,3300
TCL	True	21	1700,1800,1900
TCL	True	25	2200,2300,2400
TCL	True	34	3200,3300,3400

工作流过程有向图描述，如图 5-1 所示。

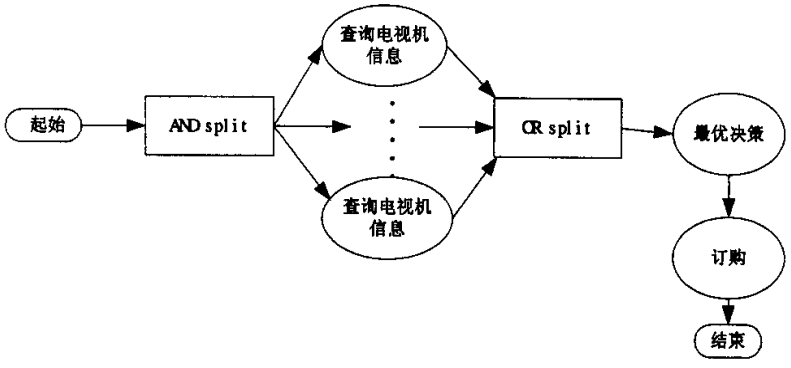

图 5-1　最优性价比电视机选购流程示意图

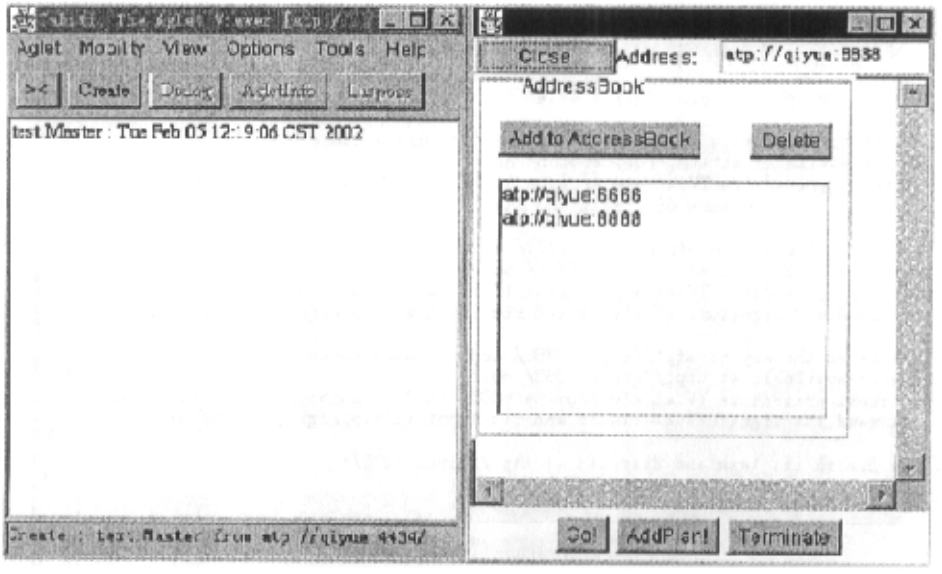

图 5-2 系统运行界面

启动 Aglets，进入系统界面。B、C 和 D 分别启动本体代理 Service，提供了访问电视机信息的 Web 服务。A 启动本地工作流管理代理 Master，Master 负责启动移动工作流代理 Slave，并保存其返回状态信息。Master 启动后的界面如图 5-2 所示。

图 5-2 左边窗口为 SAglets 运行界面其文本框显示本地代理 Master 已经启动在服务器 atp://qiyue:4434 qiyue 为 A 的机器名 4434 为端口号图 5-1 右边窗口为 Master 运行界面左上角按钮用于关闭所编辑的地址簿右上角的 Address 文本框用于输入地址信息这两个组件下面是地址簿编辑框其中的 Add to AddressBook 和 Delete 按钮用于增加和删除电视机供应商地址被地址簿编辑框遮住大部分的是另一个文本框用于显示 Slave 运行信息左下角按钮 Go! 用于启动移动代理 Slave AddPlan! 用于在移动代理 Slave 已经离开源主机后追加路由信息 Terminate 用于中断移动代理 Slave 运行。

B、C 和 D 的地址分别增加到 Master 的地址簿中在点击 Master 的 Go 按钮后移动代理 Slave 被创建并启动运行直至结束此时 Master 运行信息显示框内容如图 5-3 所示。

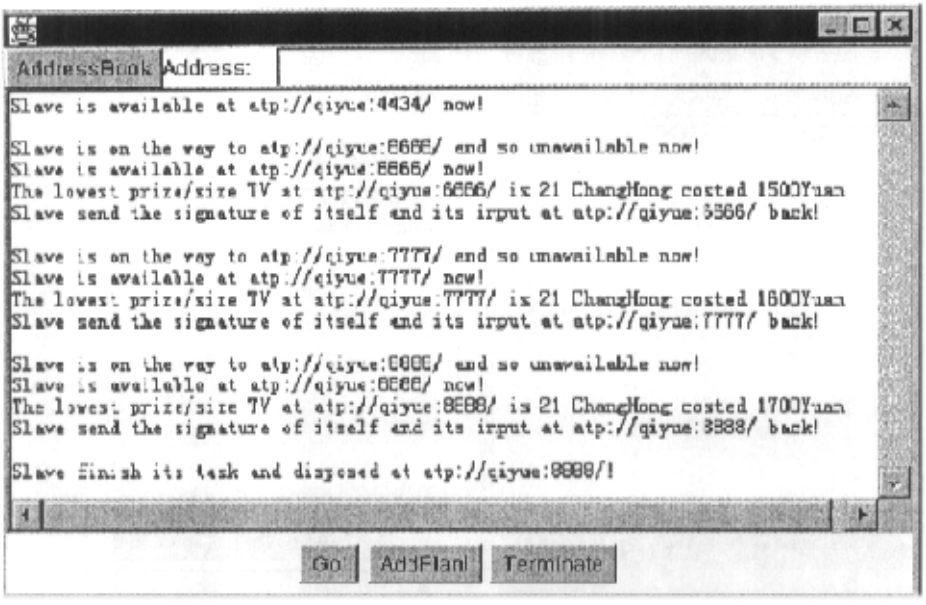

图 5-3　移动工作流代理 Slave 运行信息

5．4　相关问题讨论

协同工作引擎是个多技术结合交叉应用研究，因此在研究和实现当中，除了要面对各个研究领域独特的问题和困难之外，还要重点解决领域交叉所带来的问题。

5．4．1　系统安全性问题

安全性问题是基于移动代理技术构件的应用所面临的最棘手的问题。例如，在一个基于移动代理的电子商务系统中，保证系统和代表各种利益的代理的安全是进行正常商务活动的前提条件。如何建立一个高效的安全体系；如何保护主机不受恶意代理攻击而不过多地限制移动代理的访问权限；如何保护代理不受恶意主机地攻击；如何区分代理是善意还是恶意并避免恶意代理像病毒一样在网络上传播，肆意攻击低层通信网络和其它系统，这些问题到目前为止还没有一个完美的解决方案，仍将是今后研究的关键问题。

5．4．2　效率和性能

移动代理系统的执行效率有待提高。移动代理技术减少了中间数据的传输量，节省了网络带宽，克服了网络延迟，但增加了代码移动的额外开销，另外代理移动过程中的状态（代码、数据、运行）捕捉、序列化、安全检查、注册、恢复运行等工作也会增加额外的开销。只有在低速、高延迟、连接不可靠的网络环境中或中间数据传输量非常大时，移动代理技术才显示出效率上的优势，而在一般的分布式应用中并不比其它技术高效。

从支持平台无关性、安全性等方面考虑，几乎所有的移动代理系统都采用解释型语言。解释型语言和编译型语言相比，执行速度要慢很多，这也是移动代理系统效率不能显著提高的一个重要原因，因此提高解释型语言的执行速度是提高移动代理执行效率的一个重要手段，这方面比较突出的工作是 Java 语言的 Just-In-Time 技术。

提高效率的另一个重要手段是设计出好的移动策略，根据移动代理的任务、当前网络负载和服务器负载等外界环境，决定是否需要移动，如果需要移动则动态地为移动代理规划出移动路径，使移动代理在开销最小地情况下，最快、最好地完成其任务。

当前所有移动代理系统的容错机制都十分简单，这将严重影响移动代理系统的性能，这也是制约移动代理技术流行的重要原因之一。更高一级的容错机制应包括对用户屏蔽错误、及时发现和改进错误、确保错误发生前后系统和代理的状态一致等。

第六章 结束语

本文从分析工作流技术研究的起源开始，分析了工作流发展的现今制约其在 Web 应用领域发展的根本原因是其集中控制的体系结构。

本文提出的基于移动代理的工作流系统——协同工作引擎，结合了工作流技术、移动代理技术和 Web 服务技术三者的优势。该体系结构可以很好的解决工作流系统的底层分布运行环境，分散控制，以及动态业务重组问题。同时，采用 Web 服务技术框架作为应用集成框架结构，使得工作流系统之间以及工作流系统与其它应用系统的集成变得很容易。

本文另一个重点解决的问题是提出了一个基于工作流的描述流程语义的本体及其构造方法。协同工作引擎，本质上是一个多代理系统。因此，代理之间的协调工作是首要解决的问题。本文，采用本体技术进行过程定义的语义描述。本体的抽象采用面向对象的分析与设计方法，这使得对本体的扩充极其方便。

最后，以一个在线 Web 服务为例，详细论述了工作流建模方法。

由于本课题研究属工作流技术与移动代理、Web 服务技术等多领域交叉性研究，使得我们不得不面临来自各个研究领域以及领域交叉所带来的复杂问题。例如，安全问题，效率问题等。相信通过各个领域研究的不断发展，各种问题一定会逐渐得以解决。

进一步研究展望：

- 在本论文研究的基础上，引入强代理技术。
- 进一步对企业间交互过程进行深入细致的调研，建立较完备的企业间协同工作流过程定义。
- 继续研究代理技术、Web 服务技术的有机结合问题。
- 研究基于 Web 服务的应用集成。
- 研究流程语义问题，增强协同工作引擎自动处理能力。

参考文献

[1] J.E.White.Telescript Technology:The Foundation for the Electronic Marketplace.General Magic White Paper,1994

[2] G.Cabri,L.Leonartdi,etc. MobileAgent Tchnology:Current Trends and Perspectives.http://sirio.dsi.unimo.it/MOON/papers/aica98.ps.gz

[3] S.Bouchenak. Picking threads state in the Java system 1999.http://sirac.imag.fr/PUB/99/99-ersads-sara-PUB.ps.gz

[4] S.Bagchi,K,Whisnant,etc.Chameleon:Software Infrastructure for Adaptive Fault Tolerance.1999.http://chaos.crhc.uiuc.edu/~bagchi/Research/ipds98.ps.gz

[5] J.Bredin,R.T.Maheswaran.A Game-Theoretic Formulation of Multi-Agent Resource Allocation.2000. http://www.cs.Dartmouth.edu/~jonathan/agents/agent00.ps.gz

[6] S.K.Madria,B.Bhargava. A Transaction model for mobile computing .proc. of the 1998 international database engineering and applications sysposium,1998,pp.92-102

[7] J.Baumann.A protocol for Orphan Detection and Termination in Mobile Agent System.1997.http://www.informatik.uni-stuttgart.de/ipvr/vs/personen/../lehre/ws9899/vorlesungen/MA

[8] T.Sandholm.Approaches to winner determination in combinatorial auctions.Decision Support Systems.http://128.252.165.44/~sandholm/windetreview.ps

[9] Mike Uschold. 1998. Knowledge level modelling: concepts and terminology. The Knowledge Engineering Review, Vol. 13:1, 1998, 5-29

[10] Guarino, N. and Giaretta, P. 1995. Ontologies and Knowledge Bases: Towards a Terminological Clarification. In N. Mars (ed.) Towards Very Large Knowledge Bases: Knowledge Building and Knowledge Sharing 1995. IOS Press, Amsterdam: 25-32

[11] William, S. And Austin, T. 1999. Ontologies. IEEE Intelligent Systems, 1999 Jan/Feb: 18-19

[12] Lenat, D and Guha, RV, 1990. Building Large Knowledge-based Systems: Representation and Inference in the CYC Project. Addison-Wesley.

[13] Uschold, M, King, M, Moralee, S and Zorgios, Y. 1998. The enterprise ontology. The Knowledge Engineering Review 13(1).

[14] Fikes, R, Cutkosky, M, Gruber, T and van Baalen, J, 1991. Knowledge sharing technology

project overview, Technical Report KSL-91-71, Stanford University, Knowledge Systems Laboratory.

[15] Guarino, N., Carrara, M., and Giaretta, P. 1994. An Ontology of Meta-Level Categories. In D. J., E. Sandewall and P. Torasso (eds.), Principles of Knowledge Representation and Reasoning: Proceedings of the Fourth International Confer-ence (KR94). Morgan Kaufmann, San Mateo, CA: 270-280.

[16] Breuker, J and van de Velde, W, 1994. CommonKADS Library for Expertise Modeling, IOS Press.

[17] Workflow Management Coalition Members. Glossary - a workflow management coalition specification. Technical report, The Workflow Management Coalition, 1994.

[18] Metzger, F. J. 1996. The Challenge of Capturing the Semantics of STEP Data Models Precisely. In Pro-ceedings of Workshop on Product Knowledge Sharing for Integrated Enterprises. Basel, Switzerland, PDTAG-AM, Product Data Technology Advisory Group, ESPRIT project 9049.

[19] T.J. Mowbray and R. Zahavi. The ESSENTIAL COBRA: System Integration Using Distributed Objects. John Wiley and Object Management Group, 1995.

[20] M.R. Genesereth and R.E. Fikes. Knowledge interchange format, version 3.0 reference manual. Technical Report Logic-92-1, Computer Science Department, Stanford University, 1992.

[21] James Hender "Agents and the Semantic Web ", IEEE Intelligent Systems, vol.16-2, March-April 2001,pp.30-37

[22] Tim Berners-Lee, James Hendler, and Ora Lassila "The Semantic Web", Scientific Am, vol. 284, no.5, May 2001, pp.34-43

[23] The semantic web: The roles of xml and rdf. IEEE Expert, 15(3), October 2000.

[24] Tanja Sollazzo, Siefried Handschuh, Steffen Staab, Martin Frank, "Semantic Web Service Architecture -- Evolving Web Service Standards toward the Semantic Web"http:// www.aifb.uni-karlsruhe.de/WBS/ sha/papers/swobis_flairs.pdf

[25] Burkhard Bleckat, Nils Haberland, Stefan Puls.EAI And Web Service http://akea.iwi.unisg.ch/downloads/20020726-eai-ws.pdf

[26] [林守勋，林宗楷，郭玉钗，等.多 Agent 协同工作环境 MACE [J] .计算机学报，1998，21(2)：188-192.

[27] 陆春进、张友良、王宏典,面向并行工程的协同 CAD 系统的初步研究,机械科学与技术, 98,vol.17(6),pp1023-1025

[28] IEEE, ETSI, and Bluetooth Meeting.August 31, 1999

[29] Sumi Helal, Mei Wang and Arun Jagatheesan. SERVICE-CENTRIC BROKERING IN DYNAMIC E-BUSINESS AGENT COMMUNITIES http://www.harris.cise.ufl.edu/projects/publications/BPtemp148.pdf

[30] Web Services (2002) http://www.w3.org/2002/ws/desc/

[31] WSDL (2002) http://www.w3.org/TR/wsdl

[32] SOAP (2002) http://www.w3.org/TR/soap12-part0/

[33] 罗海滨等．工作流技术综述．软件学报．Vol．11,No．7，2000，pp899~907

[34] Blake, M.B. " Agent-based Workflow Configuration and Management of On-line Services", Proceedings of the International Conference on Electronic Commerce Research (ICECR-4), pp 567-588, Dallas, TX, November 2001

致　谢

光阴荏苒，岁月如梭，转眼间就到了毕业答辩的时候，想起攻读硕士学位学习生活期间的点点滴滴，无不让人难以忘怀！

经过近一年的课题研究和探索，论文终于完成了。在这里我首先要感谢我的导师吴跃教授的悉心指导，使我顺利克服了课题中的困难和迷茫。吴老师严谨治学的态度，孜孜不倦的工作精神使我真正体会到了研究的所需要的勤奋和执著。在此，我衷心的祝愿吴老师身体健康合家幸福！

感谢教研室的老师和师兄师姐师弟们，感谢你们给予我的无私帮助。

感谢妻子俊辉还有亲爱的家人，感谢你们对我的理解和支持。

发表论文

基于 EIP 的电子政务解决方案．计算机应用．Vol 22(10)，2002