现代集成制造系统（CIMS）系列 6

Contemporary Integrated Manufacturing Systems Series

工作流管理技术基础

——实现企业业务过程重组、过程管理与过程自动化的核心技术

Fundamentals of Workflow Management Technology

范玉顺 主编

Fan Yushun

清华大学出版社 施普林格出版社

目 录

序

工作流技术是实现企业业务过程建模、业务过程仿真分析、业务过程优化、业务过程管理与集成，最终实现业务过程的自动化的核心技术。对企业利用工作流方法进行业务过程的建模和深入分析不仅可以规范化企业的业务流程，发现业务流程中不合理的环节，进而对企业的业务过程进行优化重组，而且所建立的业务过程模型本身就是企业非常重要的知识库和规则库，可以成为指导企业实施计算机管理信息系统的模型。在深入分析企业需求基础上建立的企业业务模型可以在最大程度上提高企业实施 ERP 或者其他管理信息系统的成功率。所以，大力发展并推广工作流技术对于促进我国企业管理规范化和信息化有重要的现实意义。

工作流管理技术作为一种过程建模和过程管理的核心技术，可以与其他应用系统有效地结合，生成符合企业需求的各种业务管理系统，如办公自动化系统、项目管理软件、PDM 系统、客户关系管理系统、供应链管理系统、ERP 系统等。这些采用工作流技术作为核心开发的业务管理系统的最大特点，也是这些系统与普通的应用软件系统的最大差别，是它们具有高度的灵活性，可以按照企业的具体需求，快速灵活地生成应用软件系统，并且在客户业务过程发生变化时，迅速地进行重组来满足客户需求。

工作流管理技术的出现和迅速发展满足了企业组织结构重组与先进制造战略实施的客观需求。它的出现也促进了企业的计算机应用水平上升到一个新的阶段，即从支持企业功能实现的事务处理系统发展到支持企业实现经营目标的业务处理系统，所以，也有人将工作流管理系统称为企业的业务操作系统（BOS）。

目前，工作流技术的研究与应用在我国尚处于起步阶段，虽然已经有不少研究人员从不同的角度开展了工作流管理及其相关技术的研究，但是，有关工作流管理技术研究和应用中的许多关键技术问题还处于探索阶段。在综合了当前工作流管理技术产品情况和发展趋势情况下，结合作者几年来在工作流管理技术研究中取得的成果，对工作流管理系统的产生背景、基本概念、系统结构、实现方法、实施策略进行了全面的介绍，本书对工作流管理技术相关的研究情况和产品现状进行了深入的分析。此外，本书还对工作流管理技术的研究与应用实施中的一些关键技术问题开展深入的讨论，就工作流管理技术的企业需求、工作流建模、工作流执行、应用集成机制、过程集成方法和过程自动化、工作流技术在企业经营过程重组以及 CIMS 中的应用等问题提出了自己的见解，并详细介绍了作者设计开发的基于 CORBA 和

WEB 技术的分布式工作流管理系统 CIMFlow 的设计与实现技术。

本书为有志于从事工作流管理技术的研究人员全面了解工作流管理技术提供了详细的科研资料，是研究人员开展深入研究的基础。本书也可以作为企业实施经营过程重组和工作流管理系统的参考资料，还可以作为计算机、自动化、机械制造专业的研究生学习工作流管理技术的教材使用。

在本书的写作过程中，得到了清华大学自动化系吴澄院士的大力支持。吴澄院士不仅大力推荐了本书的出版，还对项目组开展工作流及其相关技术的研究提供了全面的指导和支持。在此，衷心地感谢吴澄院士的支持和帮助。衷心感谢德国帕德波恩大学经营计算系 Ludwig Nastansky 教授、张红博士、王蓓女士，对他们为编写本书第 5.5 节付出的辛勤劳动和在工作流技术上开展的有益讨论表示衷心的感谢。衷心感谢清华大学自动化系熊光楞教授的支持和帮助。作者还要衷心感谢作者的同事们和研究生们在工作流管理技术的研究开发上所作出的贡献和对本书的写作所提供的支持。作为我国第一本全面介绍工作流管理技术的书，作者力图客观全面地向读者介绍工作流管理这个新兴技术，书中的许多内容是作者研究开发工作的体会。由于工作流技术还正处在迅速发展阶段和作者本身的水平所限，书中的缺点和错误在所难免，欢迎广大读者批评指正。

范玉顺

2000 年 9 月于清华园

第 1 章 先进制造战略与企业计算机应用

1.1 企业经营环境的变化

自 20 世纪 70 年代以来，世界市场已经由传统的相对稳定逐步演变成动态多变的市场，企业之间的竞争也由过去的局部竞争演变成全球范围内的竞争。同行业之间、跨行业之间相互渗透、相互竞争日趋激烈。在企业间竞争日益国际化、白热化的同时，当今企业所面临的社会、经济、制造环境与客户需求也已经发生了巨大的变化。

70 年代以前, 一个产品的生命周期很长, 构成产品的技术相对比较简单。一个新产品上市, 其余工厂很快就生产出功能相同的产品, 因此, 市场竞争主要围绕如何提高生产率进行, 大批量刚性生产由此应运而生。这种生产线建筑在产品部件化、部件标准化及加工工序规范化的基础上, 然后应用泰勒的管理思想, 把工人固定在以一定节奏运动的生产线旁, 从事几项简单和熟练的加工操作, 从而大幅度地提高了劳动生产率。从 70 年代中、后期开始, 直到 80 年代, 市场竞争转到如何全面提高 T、Q、C、S 上。其含义是：T—产品的交货时间或新产品的上市时间; Q—质量; C—成本; S—售前和售后服务。进入 20 世纪 90 年代之后, 市场竞争主要是围绕新产品的竞争而展开的。这是因为, 一个新产品的价格总是高于其价值的, 通过竞争, 价格才逐渐接近价值。当一个产品失去其独占期, 就意味着这个产品生命周期的结束。因此, 今天的企业依靠一项或几项独占性技术构成的新产品就能获取高额利润。这种现象反映了这个时代的一个基本特征, 即独占性技术构成了产品的主要价值及价格。信息时代的实质是知识的时代, 大量知识产生、传播、应用, 使知识—技术—新知识诞生的周期越来越短。如何敏捷地利用技术提供的可能性及时抓住市场对新产品需求的机遇, 快速开发新产品, 已成为赢得竞争的最重要的手段。

从图 1.1 给出的变化趋势上可以看出，在当今市场环境下，产品的生命周期在不断缩短，同一种产品的重复定单数量在不断减少，每批定单中对一种产品的数量也在不断减少，而客户对产品的多样性的要求呈持续增长的趋势。

除了传统的对企业提供产品的低成本、高质量要求外，当今的企业生产经营还要能够适应以下的市场环境变化：

(1) **市场被迅速地分割：在**当今，除了极个别的产品的市场（如微软公司的 Windows 操作系统、波音飞机公司）被少数公司垄断以外，大部分产品的市场都被许多公司在动态

地分割，每个公司凭借其实力和经营策略试图占有更多的市场份额。通过垄断来独占市场或者通过大批量生产的方式获得高额利润的机会已经越来越小。相反，在市场上能够异军突起、独占鳌头的公司则是依靠推出能够领导市场潮流的创新产品，如近几年兴起的的移动通讯产品、电子商务软件、群件系统软件平台等。

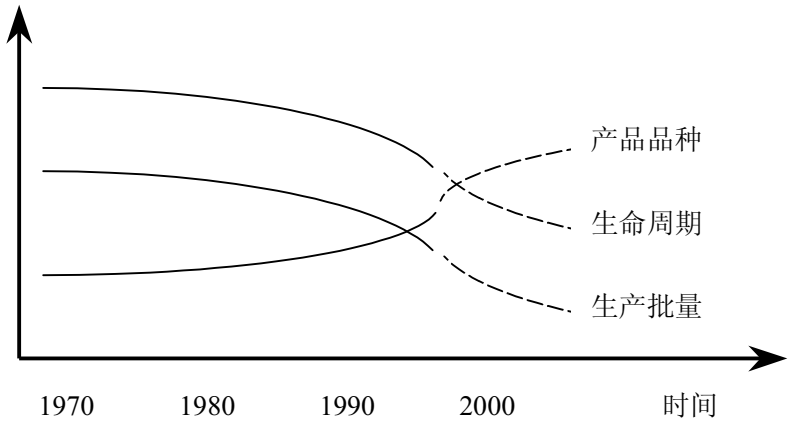


图 1.1 产品的变化趋势

(2) **产品的生命周期缩短：**与 20 世纪 70 年代相比，当今产品的生命周期已经显著地缩短，产品的更新换代速度在显著加快。一种产品从投放市场到最终被淘汰的时间已经大大缩短。由于整个产品的销售时间缩短，企业在每个产品上的利润回报时间也大大缩短了，这就要求企业在研制开发这个产品上的投资能够在短期内得到效益回报。以汽车工业为例，在 20 世纪 50-60 年代，汽车公司推出的大部分车型平均可以有 15~20 年的生命周期，而到了 20 世纪 90 年代后期，汽车的平均生命周期已经缩短到了 5~7 年。国际上某个著名的生产工程运输机械的公司为了加强其市场竞争能力，制定的其产品（工程车辆）发展战略为：3 周设计、3 个月试制与 3 年的生命周期。

(3) **产品交货期缩短：**由于最终用户对产品迅速交货的要求越来越强烈，交货期已经成为企业市场竞争力的一个重要因素。在有些情况下，用户对交货期的要求甚至超过了对价格的要求。在供应商与客户的对话过程中，经常可以听到“只要你能够在某时交货，价格贵一点没有关系”之类的说法。缩短产品交货期主要包括缩短产品的设计周期与制造周期。对于简单的装配型产品，如小家电、自行车、低档次的计算机设备，由于其设计过程相对简单，因此缩短交货期的主要问题是解决生产组织和物料及时供应问题。对于复杂的产品，如飞机、大型船舶、大型发电机组、复杂的汽车模具，缩短设计过程并且提高产品的设计质量成为缩短产品交货期的更关键的因素，而且经常是决定性的因素。对于这些复杂产品，减少由于设计不合理而造成的在产品制造过程中出

现加工困难、废品增加、返工次数增加、甚至不得不重新修改设计是缩短产品交货期的最迫切需要解决的问题。

(4) **产品的复杂性增加：**随着产品设计水平的提高和产品功能的不断增加，当今产品已经变得越来越复杂。这对企业的产品研制水平提出了越来越高的要求。以汽车为例，最早的汽车设计主要考虑机械设计问题，而当今的汽车除了机械部件越来越复杂外，还需要考虑电子点火系统、安全保护装置、空调系统、无线电系统、乃至全球定位系统。产品复杂性增加直接导致产品研制费用和产品开发周期的增加，尤其是制造用来进行试验和分析的产品原型的时间和费用迅速增加。因此，如何降低产品的研制费用，尤其是原型制造费用，是降低产品成本的关键问题。许多公司已经开始使用先进的计算机数字化产品模型与虚拟制造技术，通过在计算机上对数字化产品模型进行仿真分析，从而避免在物理上制造产品原型的方法来降低产品研制费用。典型的例子是波音公司的 777 型飞机的研制中采用了数字化产品模型和虚拟制造技术，该公司在没有生产物理样机的情况下完成了 777 型飞机的设计开发，并成功的将该机型投入了市场销售。

(5) **定制产品数量增加：**当今客户已经越来越不满足于大众化的通用产品，他们希望制造企业能够按照他们特定需求生产符合其特定功能要求、环境要求、个人兴趣、甚至审美观念的产品。按照特定客户需求设计制造的产品称为定制产品。用户不仅可以对定制产品提出需求，甚至可以直接参与其设计过程，对设计方案提出他们的建议。目前，客户定制产品的种类与在企业整个产品数量中的比例呈不断上升的趋势。许多大的轿车生产厂家已经推出了网上客户定制服务软件，供用户对他想购买的轿车提出要求。另外，国外的糖果厂还提供按照客户需求定制礼品巧克力、礼品饼干、蛋糕的业务。同样，在自行车、冰箱、家具等行业，提供定制服务的公司数量也在迅速增加。提供定制服务已经被认为是企业扩大其市场份额的一个重要手段。

(6) **环保要求日益增强：**随着全球环境保护呼声的日益高涨，对制造企业的环境要求也越来越强烈。对制造企业在环境保护方面的要求主要包括两大方面：一是对企业在产品制造过程中对环境的低污染要求，包括噪音、废气、废水、废液、废料；另外一方面指产品在报废以后要对其大部分的部件尽可能地回收重用，对于不能够回收重用的要能够方便地进行分解处理，从而在最大程度上减少对环境的污染。近些年来，与环保要求相关的绿色制造、多生命周期产品设计、可拆卸设计技术得到了高度的重视。

(7) 劳动力成本提高与工作时间缩短：虽然还有相当多的不发达地区有廉价的劳动力，但是对于许多现代企业有技术人员还是相当紧缺，而且技术工人的劳动力成本也在呈不

断上升的趋势，这就导致了企业产品成本的增加。另外随着国家、地区公共节假日的增多，劳动保护法条例的不断细化和贯彻执行，劳动力的工作时间还在不断地缩短，这也会导致企业的劳动力成本提高。

以上的市场环境变化因素对企业的生产经营提出了更高的要求，而且这些不同的因素相互之间又是矛盾的，如产品复杂性增加与按照客户需求定制生产直接导致产品成本增加、生产周期与交货期延长，劳动力成本的增加和企业满足环保要求都将导致产品成本的增加。因此，虽然全面提高企业的 T（交货期短）、Q（高质量）、C（低成本）、S（更好的服务）水平是企业努力的方向，但是要真正实现这些目标并不是一件容易的事情。企业必须综合利用各种先进制造技术，在网络与信息技术的支持下，改进现行的生产经营模式和组织结构，才有可能全面实现这些目标，并在未来的市场竞争中赢得更多的份额。

1.2 先进制造战略

制造业是国民经济的支柱产业，无论是在发达国家、还是在发展中国家，制造业在国民经济中均占据主要地位。1994 年，在新加坡召开的国际生产工程学会（CIRP）第 44 届年会上，大会主席日本吉川教授在总结报告中指出，世界上各个国家经济的竞争主要是制造技术的竞争，在各个国家的企业生产力构成中，制造技术的作用一般占 60%左右。由此可见，制造技术水平的高低已成为衡量一个国家经济实力和科技发展水平的重要标志之一。

在当前竞争激烈、需求变化非常迅速的市场环境下，传统的制造业正在发生着深刻的变革，先进制造技术正显著地提高企业的产品质量、经济效益和市场竞争力。越来越多的企业将先进制造技术作为企业适应迅速多变的市场需求和提高竞争力的主要手段。先进制造技术还在大幅度的改善企业产品结构、生产过程和经营管理模式上发挥重要的作用。越来越多的企业把能够高质量、快响应、灵活、敏捷地满足客户需求的先进制造技术作为企业继续生存并保持发展的有效途径。

先进制造技术是传统制造业不断吸收机械、电子、信息、材料、能源及现代管理等技术成果，并将其综合应用于制造全过程，以实现优质、高效、低消耗、清洁、灵活生产，从而取得较理想的技术与经济效果的制造技术的总称。先进制造技术是制造技术的最新发展阶段，它既是由传统制造技术发展而来，又随着高新技术的引入和制造环境的变化而产生了质的飞跃，现已经成为能够有效地控制制造系统中的物料流、信息流和资金流的工程技术。作为面向 21 世纪的制造技术，先进制造技术具有以下显著的特点：

(1) **先进制造技术贯穿了制造的全过程：**传统制造技术一般仅指加工制造的工艺方法，实际上只是制造全过程中的一部分；而先进制造技术贯穿了从市场预测、产品设计、采购、生产经营管理、制造装配、质量保证、市场销售、售后服务、报废处理、甚至回收再利用等整个制造过程，使整个制造过程成为集市场、产品、制造为一体的大系统。

(2) **先进制造技术注重技术、管理、人员三者的有机集成：**实践证明，先进制造技术不是一个单纯的技术问题，在实施先进制造技术的过程中，如果只采用从技术到技术的作法，而忽略组织、管理和人员的作用，在实施后往往不会取得令人满意的效果。因此，在实施先进制造技术的过程中，要注重技术、管理、人员三者的有机集成，使制造全过程能够达到优化运行，使组织管理模式更加灵活合理。当今一些先进制造技术，如计算机集成制造、并行工程、准时制造、全面质量管理、精良生产、敏捷制造等都体现了技术与管理、人员三者紧密结合这一特点。

(3) **先进制造技术能有效支持可持续发展战略：**资源、环境与人口是当今社会面临的三大主要问题。联合国从人类长远生存的角度出发，提出了促进世界经济发展的可持续发展战略。对于制造业，先进制造技术领域中的绿色制造、清洁制造、生态工厂等新概念、新哲理，是支持可持续发展战略的有力武器，是企业跨世纪的发展战略。在企业中运用它们会有助于保护自然环境，最大限度地提高资源利用率和改善员工的工作环境。

(4) **先进制造技术是多学科交叉融合的产物：**传统制造技术的专业、学科单一，不同专业之间的界限分明，而先进制造技术的专业、学科之间不断渗透、交叉、融合，其界限逐渐淡化甚至消失，这使得先进制造技术趋于系统化、集成化，并发展成为集机械、电子、信息、材料和管理等技术于一体的新兴交叉技术。如快速原型制造技术，它可以自动而迅速地将设计思想物化为具有一定结构与功能的原型或者实际零件，从而对产品设计进行快速评价、测试、改进，以迅速响应市场需求。快速原型制造技术是机械工程、CAD、数控、激光以及材料等技术的交叉与优化集成。

由于先进制造技术不是一项或者几项单纯的具体技术，而是由传统的制造技术与以信息技术为核心的现代科学技术相结合而产生的一个完整的高新技术群，因此，可以称为先进制造技术的新技术很多，而且还会不断的出现许多新的先进制造技术。为了给读者一个关于先进制造技术的初步印象，本节介绍几个比较有影响的先进制造战略。

1.2.1 并行工程

在企业的以功能为核心划分的部门组织结构下，传统的产品研制是一种串行的过程，即在前一个部门完成某个功能后将设计结果移交给下一个部门进行下一步的设计工作。这种串行的产品研制方式存在的严重的不足是不同的设计开发部门之间缺乏交流，每个部门都仅考虑其局部的功能性能要求，从设计时间和成本上进行局部的优化。但是由于每个部门在进行局部优化的过程中未考虑后续部门的要求和产品设计制造周期中的各种因素，将导致最终设计出来的产品的可制造性和可装配性差，产品制造周期长、废品率高、产品成本高。从决策空间的角度来看待产品研制过程可以更好的理解串行设计方式存在的不足，在产品设计的早期阶段，产品设计的决策空间很大，但是随着设计过程的深入，产品设计空间越来越小，因此在产品设计的早期阶段如果不考虑后续设计过程的约束，则在产品研制的后期，留给设计者的余地就很小了，有时甚至无法完成设计制造工作。在这种情况下，经常需要从头开始修改设计，而从头开始修改设计将导致所有部门的工作全部需要重做，这不仅造成设计周期的延长，而且导致大量设计精力的浪费，并直接引起产品研制成本的增加。

为了解决串行设计存在的严重不足，1998 年美国国防分析研究所（IDA）的 R. I. Winner 提交了题为“并行工程在武器系统采购中的作用”的研究报告中提出了并行工程的概念。并行工程的定义是：集成地、并行地设计产品及其相关的各种过程（包括制造过程和支持过程）的系统方法。这种方法要求产品开发人员在设计一开始就考虑到产品的整个生命周期（从概念形成到产品报废）中的所有因素，包括质量、成本、进度计划和用户需求。

根据上述定义，不难看出，并行工程这种先进制造模式的技术核心除了继承早期计算机集成制造系统（CIMS）的信息集成外，其主要特点是过程集成。所谓过程集成是指利用计算机集成支持软件工具高效、实时地实现企业事务处理系统间的数据、资源的共享和应用间的协同工作，将一个个孤立的应用集成起来形成一个协调的企业信息系统。实现过程集成后，就可以方便地协调各种企业应用系统的功能，把人和资源、资金及应用合理地组织在一起，获得最佳的运行效益。并行工程中的过程集成主要完成产品开发过程中不同的设计活动和过程的重组，将产品开发过程中各种串行过程尽可能多地转变为并行过程，使设计人员在设计的早期阶段就能考虑到产品的可制造性、可装配性、可测试性、可维护性、可靠性及低成本等因素，促进产品设计不同阶段工作之间的及时交流、协调，避免跨不同设计阶段的大返工。

图 1.2 给出了两种不同设计模式的比较。

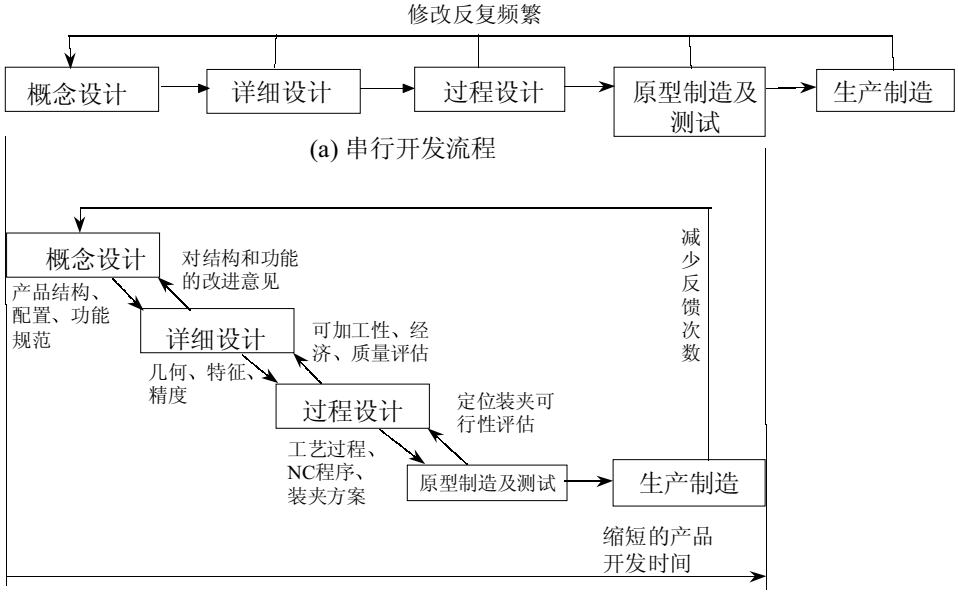


图 1.2 两种产品开发流程的比较

并行工程作为实现产品高质量、低成本、上市快及满足用户需求的先进制造技术，它基于传统的 CAD、CAPP、CAM 技术，采用先进的 PDM（产品数据管理）作为支撑平台，通过组建多学科的团队和设计流程重组，从根本上改变传统的产品设计模式。并行工程概念和方法的内涵是十分丰富的。首先，并行工程是一种集成化、系统化的设计方法学， 它强调产品全生命周期－－市场需求分析、产品定义、研究开发、设计、制造、支持（包括质量、销售、采购、发送、服务）及产品报废等各个阶段过程的集成、并发与优化。并行工程要求人们在进行产品生命周期的上游阶段的工作时，要充分考虑下游工作的可实现性。其次，并行工程强调过程重组与流程改进，并提出了以产品为中心，组成有关部门代表参加的多学科小组（群）进行“团队”工作的工作方式。这种工作方式不仅加强了部门间的协调，而且集成了多学科人员的智慧。另外，并行工程还十分重视采用先进的信息技术和产品设计技术，如采用产品数据管理（PDM）作为支持产品开发的支撑环境，采用工作流管理作为支持设计流程重组与流程管理的工具，采用产品的数字化定义、DFA（面向装配的设计）、DFM（面向制造的设计）、QFD（质量功能配置）等技术提高产品设计水平。

并行工程实施中涉及许多将影响该战略最终是否获得成功的关键技术问题，主要包括：

(1) **并行产品开发过程建模、仿真与优化：**并行工程与传统的设计方法的本质差别在于它把产品开发的各个活动作为一个集成的、并行的产品开发过程，强调下游过程在产品开发的早期参与设计，因此对产品开发过程进行建模，在过程仿真与优化分析的基础上实现对产品开发开发过程的有效管理和控制，有利于不断的改善产品的开发过程。

(2) **并行工程的集成产品开发团队的组织：**在并行工程中，产品开发由传统的部门制或者专业组变为以产品（型号）为主线的多功能集成产品开发团队。根据产品复杂程度的不同和涉及的范围不同，团队可以有企业级、产品级、部件级和零件级等不同层次。合理地组织团队，赋予团队相应的权力、责任，为团队开展工作和实施集体决策提供协同工作的环境，是实现集成产品开发重要的组织保证。

(3) 并行工程支撑环境的建立：在并行工程产品开发模式下，产品开发是由分布在异地的采用异种计算机软件工作的多学科小组完成的，多学科小组之间及多学科小组内部各组成人员之间存在着大量相互依赖的关系，并行工程的不同应用软件之间也存在着大量的数据共享和应用活动逻辑之间的约束。因此，建立能够屏蔽硬件、软件系统的异构性，并能够高效实时地支持数据管理、应用管理、应用协调和过程管理的支撑环境对于成功实施并行工程具有非常重要的意义。

(4) **数字化产品建模与 CAx/DFx 使能工具：**建立产品全生命周期的数字化模型，特别是基于 STEP 的特征模型，是高效快速的实现不同设计软件工具之间的信息交换的基础，它也是完成在计算机模拟仿真分析的基础。除了基础的支撑软件环境外，实施并行工程还需要许多计算机辅助设计制造的使能工具的支持，如 CAD（计算机辅助设计）、CAE（计算机辅助工程）、CAPP（计算机辅助工艺）、CAM（计算机辅助制造）、CAFD（计算机辅助工装系统）、DFA（面向装配的设计）、DFM（面向制造的设计）、MPS（加工过程仿真）等。

自 20 世纪 80 年代以来，并行工程在国际上得到了广泛重视，在许多应用中取得了成功。如波音飞机公司在其波音 777 型飞机的设计中采用了并行工程技术，从 1990 年 10 月开始到 1994 年 6 月结束，波音公司仅用了 3 年零 8 个月的时间就完成了波音 777 飞机的设计开发，而通常开发同样型号的飞机需要 7~10 年的时间。在波音 777 开发过程中，波音公司对产品的设计开发过程进行了重组，按照功能组建了 200 多个设计/制造团队，并实现了这些集成产品团队（IPT）的协同工作。通过全面采用三维实体建模和 100％的数字化预装配，从而将设计更改、错误或返工报废减少了 50％。成功利用并行工程技术开发的产品还有瑞士 ABB 公司的火车运输系统、美国 Chrysler 公司的新型汽车、德国西门子公司自动化部的汽车电子系统、美国洛可希德导弹与空间公司的 Thaad 导弹等。

在 863/CIMS 主题的支持下，我国于 1995 年 7 月开展了 863/CIMS 关键技术攻关项目－－并行工程。由清华大学、华中理工大学、北京航空航天大学、上海交通大学、航天总公司二院等 5 个单位联合开展技术攻关，并以航天二院某型号的复杂结构件为背景进行了初步

的应用，取得了较好的攻关成果。目前，作为并行过程攻关成果的应用推广，国内 9 家企业正在进一步开展实施并行工程的工作。实践证明，并行工程是一种加速企业新产品开发，赢得市场竞争的有效的先进制造模式与技术。

1.2.2 敏捷制造

1. 敏捷制造提出的背景

1991 年，由美国国防制造技术计划秘书办公室资助，由里海(Lehigh)大学的亚柯卡(Iacocca)研究所主持了一项关于 21 世纪发展战略的研究，这项研究由亚柯卡研究所和美国 13 家大型企业的行政首脑组成核心组进行，并邀请 100 多家企业和著名的咨询公司参与研讨，历时半年，形成了一份名为《21 世纪制造企业发展战略》的著名报告。 敏捷制造(Agile Manufacturing)正是在这份报告中总结经济发展现状、展示未来而提出的一种先进制造技术。提出敏捷制造的直接原因是市场的需求；同时，技术的发展又使其成为可能。

在 21 世纪，技术的发展及世界市场的竞争，无疑会沿着 20 世纪 90 年代展开的道路前进: 危机与机遇并存，竞争更加激烈，企业面临更加严峻的挑战。随着生活水平的不断提高，人们对产品的性能和质量的要求也越来越高。新技术的迅速涌现使得一个产品的生命周期越来越短，产品的技术含量越来越高，产品的生产批量却越来越少。用户需求日趋多样化、个性化，所有企业都将处于一种连续改变、不可预见的市场环境中。如何对现有企业进行改造与结构调整，使之能在复杂多变的环境中赢得竞争，求得生存和发展，已成为全球企业界所共同关心的问题。问题的核心是能否抓住机遇，快速响应市场，开发出新产品。敏捷制造的思想正是在这种市场竞争的背景下提出的，它是总结和预测经济发展客观规律的产物，决不只是一个新名词的问题。目前，国内外均有不少企业自觉或自发地运用敏捷制造的思想，从而取得了相当大的成功。

从制造业技术发展的角度来看，敏捷制造也是自动化技术、计算机应用技术、信息技术、网络通信技术发展的一种必然结果。CIMS 技术是采用信息集成的方式来解决设计、管理和加工制造中大量存在的自动化孤岛问题，解决其信息的正确、高效共享和交换。信息集成技术更进一步提高了生产线的产品生产效率，并使之具有柔性。在信息集成的基础上，为了进一步实现设计过程的优化和设计制造资源的优化，CIMS 技术必然向过程集成（如并行工程）和企业集成（如敏捷制造）延伸。

2. 敏捷制造的概念和内涵

在《21 世纪制造企业发展战略》报告中, 明确提出了敏捷制造的概念。其基本思想是: 通过将高素质的员工、动态灵活的虚拟组织机构(Virtual Organization)或动态联盟、先进的柔性生产技术进行全面集成, 使企业能对持续变化、不可预测的市场需求作出快速反应, 由此获得长期的经济效益。它强调人、组织、管理、技术的高度集成, 强调企业面向市场的敏捷性(Agility)。图 1.3 形象地体现了敏捷制造的概念。

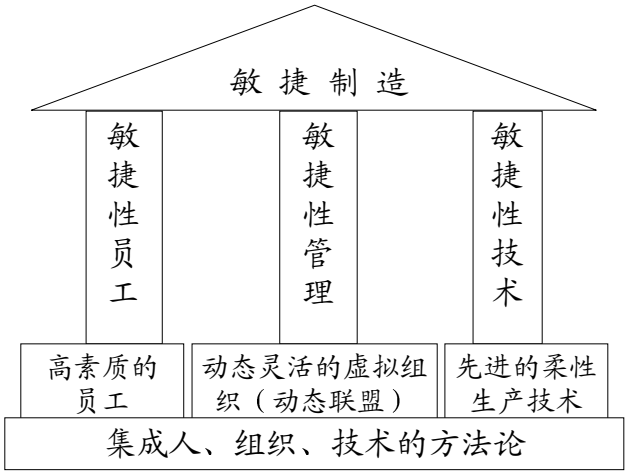


图 1.3 敏捷制造的概念

敏捷制造包含如下丰富的内涵:

(1) 敏捷制造的出发点是多样化、个性化的市场需求和瞬息万变的经营机遇, 是一种“定单式”的制造方式。

(2) 敏捷性反映的是制造企业驾驭变化、把握机遇和产品创新的能力。

(3) 敏捷制造重视充分调动人的积极因素, 强调需要有知识、精技能、善合作、能应变的高素质员工, 充分弘扬人机系统中人的主观能动性。

(4) 敏捷制造不采用以职能部门为基础的静态结构, 而是推行面向产品过程的小组工作方式, 企业间由机遇驱动而形成动态联盟。

动态联盟是敏捷制造的基本组织形态, 其含义是指企业群体为了赢得某一机遇性市场竞争, 围绕某种新产品开发, 通过选用不同组织/企业的优势资源, 综合成单一的靠网络通讯联系的阶段性经营实体。动态联盟具有集成性和时效性两大特点。它实质上是不同组织/企业间的动态集成, 随市场机遇的存亡而聚散。在具体表现上, 结盟的可以是同一个大公司的不同组织部门(以互利和信任为基础, 而非上级意识), 也可以是不同国家的不同公司。动态联盟的思想基础是共同赢(Win-Win), 联盟中的各个组织/企业互补结盟, 以整体优势来应付多变的市场, 从而共同获利。

敏捷制造作为一种新的制造哲理, 有许多新的制造思想。但是, 必须指出, 敏捷制造并

不意味着需要高额的投资作为前提, 也不需要抛弃所有过去的生产过程和结构, 而是强调如何利用旧的、可靠的生产过程和生产要素(尽可能少添加新过程)来构成新系统, 生产出更多的新产品。动态联盟就是利用已有的社会技术基础, 通过重组来实现敏捷制造的有效方式。

3. 企业组织的敏捷性

实现敏捷制造的关键是对制造企业进行敏捷化改造或重组, 增强企业的敏捷性。企业的敏捷性是指可重构(Reconfigurable)、可重用(Reusable)和可扩充(Scalable)的特性, 即 RRS 特性。对企业敏捷性的综合度量, 可以用成本(Cost)、时间(Time)、健壮性(Robustness)和自适应范围(Scope of Change)这四个度量指标。其中, 自适应范围是敏捷性的精华, 它不是预先按规定的需求范围建立某过程, 而是使企业从组织结构上具有 RRS 特性, 是企业预计完成变化活动的能力，与创新性密切相关。成本是指完成敏捷变化(或转换)的成本，变化成本与新产品的上市成本密切相关，也与产品过程的设计与实现相关。健壮性是指敏捷变化过程的坚固性和稳定性, 它与新产品和过程的质量密切相关。这四个方面相互约束, 相互关联。敏捷性必须在此四个方面取得均衡。例如, 无成本约束的快速变化是无意义的; 不健壮的变化也不实用; 自适应变化范围必须与快速的时间、合理的成本、保险的健壮性相联系。企业实现敏捷性转化的费用直接反映在产品成本、质量、市场响应之中。

从企业的组织管理角度可以用五个主要变量或要素来表示企业的敏捷性, 即通讯连通性(Communication Connectedness)、跨组织参与性(Interorganization Participation)、生产灵活性(Production Flexibility)、管理相关性(Management Involvement)和雇员使能性(Employee Empowerment)。其中，通讯连通性是指敏捷企业中跨越公司和组织机构范围的通讯、交流程度，它包括通讯技术和通讯行为两方面。跨组织参与性是指敏捷企业跨越公司范围组织进行经营活动的能力和水平。跨组织的共同参与有多种方式，如基于顾客（供应商）长期合同建立的供应链关系、合资联合体、虚拟伙伴合作关系等。在敏捷制造中主要关心虚拟伙伴合作关系，即动态联盟方式。合作伙伴间的相互信任和依赖是跨组织间的共同参与取得成功所必不可少的因素。生产灵活性是指企业的制造能力、生产及时性和可重构性的水平。敏捷企业要利用各个层次（公司层、工厂层、单元层、设备层等）的灵活性，尤其要考虑公司层的灵活性。管理相关性是指企业主动参与管理以调整其组织和资源使得敏捷制造得以实现的水平。它与企业的组织和资源、企业创新性和主动性、动态联盟的作用密切相关，它直接影响着公司及其经营过程的可重构性和灵活性。雇员使能性指企业使每个雇员能够根据公司目标定义其义务责任，进行影响其工作领域乃至整个公司的决策的能力水

平。雇员使能性包括公司建立多功能项目组的能力、对雇员的全面培训、雇员被授权和解决问题的水平等方面。

4. 实现敏捷制造的关键技术

实现敏捷制造应有哪些支撑技术呢? 在《21 世纪制造企业发展战略》报告中, 描述了美国敏捷制造企业模式中所包含的 20 个技术使能子系统。在此不一一列举, 只就其中最为重要的, 也是现有企业在敏捷化变革中首先要解决的技术问题简述如下:

(1) 基础技术---CIM 技术

CIM 技术是一种组织、管理与运行企业生产的技术, 它藉助计算机硬软件, 综合运用现代管理技术、制造技术、信息技术、自动化技术、系统工程技术等，将企业生产全过程中有关人、技术、经营管理三要素及信息流、物料流有机地集成并优化运行，以实现产品高质、低耗、上市快，从而使企业赢得市场竞争。它为实现敏捷制造的集成环境打下了坚实的基础，是敏捷制造的基础技术。

(2) 环境技术---网络技术

实现敏捷制造, 企业需要具有前述的通讯连通性, 网络环境是必备的，并应按照企业网---全国网---全球网的步骤建立, 实施企业的网络技术。利用企业网实现企业内部工作小组之间的交流和并行工作, 利用全国网、全球网共享资源，实现异地设计和异地制造，及时地、最优地建立动态联盟。基于网络的企业资源计划管理系统和商品供应链系统都将为敏捷制造的实施提供必需的信息。

(3) 统一技术---标准化技术

以集成和网络为基础的制造离不开信息的交流，交流的前提是有统一的交流规则，这是标准化的工作。执行电子数据交换标准 EDI、产品数据交换标准 STEP 及超文本数据交换标准 SGML 等, 是进入国际合作大环境、参加跨国动态联盟的前提。

(4) 虚拟技术---模型和仿真技术

敏捷制造通过动态联盟和虚拟制造来实现, 因而对产品经营过程进行建模和仿真, 采用基于仿真的产品设计和制造方法是十分必要的。另外, 作为敏捷制造在产品设计和制造过程中的主要手段之一的虚拟原型系统, 也是以模型和仿真技术为基础。

(5) 协同技术---并行工程技术

并行工程技术是对产品及相关过程(包括制造过程和支持过程)进行并行、一体化设计的一种系统化技术。该技术要求产品设计人员在设计一开始就考虑到产品全生命周期（从

概念形成到产品报废）中的所有因素，包括质量成本、速度、进度及用户要求等。并行工程通过组成多学科的产品开发小组协同工作，利用各种计算机辅助工具等手段，使产品开发的各阶段既有一定的时序又能并行。同时采用由上、下游因素共同决策产品开发各阶段工作的方式，使产品开发的早期就能及时发现产品开发过程中的问题，从而缩短开发周期，降低成本，增强对市场的响应敏捷度。

(5) 过程技术—工作流管理技术

动态联盟是面向具体产品而动态创建的虚拟公司，其组织结构的临时性和动态性，加上产品研制过程的创新性和协同特性，在很大程度上决定了动态联盟的管理将采用或者基于项目管理的方式来进行。能够有效支持企业业务重组、业务过程集成、项目管理和群组协同工作的工作流管理技术对于实施动态联盟具有重要的支持作用。另外，工作流管理系统还可以作为企业间信息集成的使能工具，基于 WEB 和基于邮件方式的工作流管理系统可以为企业灵活地组建动态联盟和实现信息交换发挥重要作用。

此外，集成框架技术、集成平台技术、数据库技术、决策支持系统、人机工程、人工智能等也都是支持敏捷制造和动态联盟的重要技术。敏捷制造是 21 世纪企业生存、竞争的必要条件，它表示的是一种在不可预见、持续变化前提下的适应和驾驭市场变化的能力，它要求整个社会的协同努力。

1.2.3 虚拟制造技术的发展

1. 虚拟制造的定义及内涵

虚拟制造(Virtual Manufacturing，VM)是 20 世纪 90 年代发展起来的一种先进制造技术，也是敏捷制造的一种实现手段。国内外的许多学者对虚拟制造进行过定义，比较有代表性的定义如下:

虚拟制造是实际制造过程在计算机上的本质实现，即采用计算机仿真与虚拟现实技术，在计算机上群组协同工作，实现产品的设计、工艺规划、加工制造、性能分析、质量检验以及企业各级过程的管理与控制等产品制造的本质过程，以增强制造过程各级的决策与控制能力。

虚拟制造是一种通过计算机虚拟模型来模拟和预估产品功能、性能及可加工性等各方面可能存在的问题，提高人们的预测和决策水平，从而使制造技术走出主要依赖于经验的狭小天地，发展到全方位预报的新阶段的先进制造技术。图 1.4 表明了虚拟制造和实际制

造的关系。

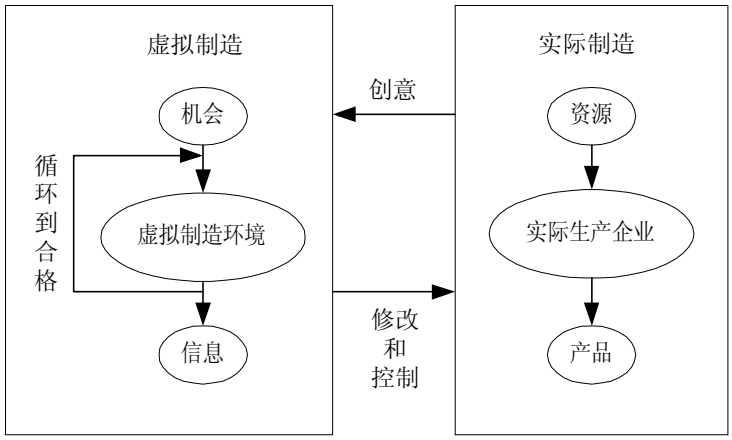


图 1.4 虚拟制造与实际制造

(1) 与实际制造相比，虚拟制造的主要特点是：产品与制造环境是虚拟模型，在计算机上对虚拟模型进行产品设计、制造、测试，甚至设计人员或用户可“进入”虚拟的制造环境检验其设计、加工、装配和操作，而不依赖于传统的原型样机的反复修改；还可将开发的产品（部件）存放在计算机里，不但大大节省仓储费用，更能根据用户需求或市场变化快速改型设计，快速投入批量生产，从而能大幅度压缩新产品的开发时间，提高质量、降低成本；

(2) 可使分布在不同地点、不同部门的不同专业人员在同一产品模型上同时工作，相互交流，信息共享，减少大量的文档生成及其传递的时间和误差，从而使产品开发以快捷、优质、低耗响应市场变化。

2. 虚拟制造系统及基本要求

虚拟制造系统(Virtual Manufacturing System, VMS)是基于虚拟制造技术实现的制造系统，是现实制造系统(Real Manufacturing System, RMS)在虚拟环境下的映射，VMS 生产的产品是可视的虚拟产品，是一个数字化产品，它具有真实产品所必须具有的特征，并具有动态结构及决策、控制、调度、管理等四个机制。虚拟制造技术和虚拟制造系统涉及到整个产品开发和制造过程的方方面面。对于产品来说，涉及到整个产品生命周期的各个方面，对于制造过程来说，涉及到整个工厂生命周期的各个方面。

对 VMS 的基本要求体现在从 RMS 到 VMS 的映射中，应反映出 RMS 的要求，因此下列要求必须得到满足:

(1) VMS 与 RMS 具有功能一致性。

(2) VMS 与 RMS 具有结构相似性。

(3) VMS 的组织与实现具有高度的柔性。

(4) VMS 应实现“全集成”，即信息集成、智能集成、串并行工作机制集成、人机集成、过程集成、资源集成及技术集成等。

(5) 实现虚拟环境下分布式并行处理的智能协同求解和虚拟环境下系统全局的最优决策。

(6) VMS 在虚拟环境下进行，其工作过程及各阶段的运行状态和结果都应该具有可视性，对其整个制造过程能进行仿真。

3. 虚拟产品开发及关键技术

虚拟产品开发(Virtual Product Development, VPD)是虚拟制造研究领域中的一个重要内容，是以计算机仿真、建模为基础，集计算机图形学、智能技术、并行工程、虚拟现实技术和多媒体技术为一体，由多学科知识组成的综合系统技术。它是现实产品开发环境下的映射。它将现实产品开发环境和全过程的一切活动及产品演变成基于数字化的模型，对产品开发的行为进行预测和评价。应用虚拟现实技术，可以达到虚拟产品开发环境的高度真实化，并使之和人有着全面的感官接触和交融。

虚拟产品开发将从根本上改变设计、试制、修改设计和规模生产、维护的传统模式。在产品真正制造和销售之前，首先在虚拟开发环境中生成数字化产品原型或软产品原型(soft prototype)代替传统的硬样件 (hard prototype)，进行测试，对其性能、可销售性、可维护性、成本和外观等进行预测和评价，从而缩短产品开发周期，降低开发成本，提高企业快速适应市场变化的能力。

虚拟产品开发的关键技术如下：

(1) 产品开发的过程建模

在并行工程指导下，产品开发过程是多学科群体在计算机网络通讯环境的支持下，在开发时间和资源约束下，建立产品全生命周期信息的开发组织结构和开发任务的动态调控流程。过程建模应考虑的内容有：

① 过程模型---过程模型的描述方法及在计算机上的处理和实现，过程的动态调整和优化。

② 组织模型---规划和描述组织结构、活动分工、权限和责任定义。

③ 资源模型---对信息、设备、人力、资金等资源进行动态规划和配置控制。

④ 约束规则---时间约束是首要约束，在资源允许下，增大产品开发活动的并行度；

功能约束反映开发活动中后续活动对前期活动的功能要求；关键参数约束是开发活动之间及开发活动中单元之间相关性和一致性的保障。

⑤ 过程监控和协调---进行实际约束管理、过程的实时监控调度和冲突仲裁，保障过程按照最优方向进展。

(2) 支持 VPD 的产品数据模型

产品建模主要以并行设计及面向对象技术为工具,以 STEP 标准为思想, 建立基于装配的约束参数化的特征产品定义模型。其特点如下:

① 具有统一的数据结构

产品采用百分之百的数字化定义方式, 使产品数据模型在产品开发的各个阶段无二义性。

② 并行方式定义产品

并行产品定义是一种系统工程方法, 它使开发人员一开始就考虑到产品生命周期里的所有环节。面向装配的设计(Design for Assembly)、面向制造的设计(Design for Manufacturing)、面向测试的设计(Design for Testing)、甚至考虑到产品报废后的可处理性等绿色产品设计思想等在并行方式的产品定义中可得到良好的体现。

③ 支持工程分析工具的应用

工程分析工具利用已经建立好的产品模型, 对零部件甚至整机进行有限元受力分析、热应力分析以及运动仿真、性能仿真、装配仿真，仿真的结果可直接用于指导数字化产品的设计修改，不必通过实物模型来验证。

④ 支持产品异地、并行设计

由于产品模型在计算机上定义，加之网络通讯的迅速发展，处于异地的设计人员也可方便地进行交流。设计队伍中除含有设计、制造、装配、试验等专业人员外，还有合作伙伴、用户代表等，这样在产品开发过程中能及早地发现问题，并使这些问题在产品开发的早期阶段就得到解决。尽量避免下游重大问题的反馈所造成的时间拖延、成本上升等现象。

(3) 多种协同机制

为了实现原型系统中各群体、各层次对信息、资源的共享，协调处理各种更新、冲突和竞争，必须具有各种协同机制。

① 组织协同

在虚拟产品开发体系下，是以团队(Teamwork)的工作方式进行的。这就存在小组(Team)内的人员协作和联盟内各小组之间的协作。小组内的各成员发挥自己的专长，通过自己管

理、员工参与和适宜的合作机制，实现单元小组的最大效能。小组间的协同则依靠动态联盟的组织机制，保证产品开发的高效和高质量。

② 人机协同

在系统设计方法中考虑：感知层面上采用人机联合感知，思维层面上采用人机共同分析，决策层面上采用人机相互协作，充分发挥各自优势。在系统实现上，考虑系统的适度自动化和人机的合理分工，特别在智能控制部分，要根据各环节功能和环境确定“机主人辅”、“人主机辅”或“人机耦合”等控制策略。要强调人机协作来表现高质量的产品开发，而不是消除人的因素来达到全盘自动化。

③ 知识协同

虚拟产品的开发涉及许多技术和环境的基础数据库和知识库。制造全球自动化的趋势，也要求各种产品开发信息和知识的协同、合作和共享。在产品的开发过程中必须解决数据的更新、版本的升级和知识间的冲突，因而必须有一个机制保障这些数据库和知识库间的协调。

④ 人、技术、组织三者的协同

在产品开发系统中，技术是基本手段，人是主体，组织是反映开发活动中人与人的相互关系。一个真正经济、高效、安全的虚拟产品开发系统，应该是由技术、人和组织三方面高度集成和协同的系统，它既是一个技术系统，也是一个社会系统。

(4) 全生命周期的产品演变仿真和产品开发全过程的活动仿真

全生命周期的产品演变仿真是通过产品的数字模型，反映产品从无到有，再到消亡的整个演变活动，用户和开发者在制造实物之前即可评审其美观、可制造性、可装配性、可维护性、可销售性和环保性能等，从而确保产品开发的一次成功率。

产品开发全过程的活动仿真旨在通过开发活动的数字模型，反映动态联盟组织形式下，产品开发活动的功能行为和运作方式，仿真虚拟产品开发的设备布置、物流系统、资源的利用和冲突以及组织结构、生产活动和经营活动等行为，从而确保产品开发的可能性、合理性、可靠性、经济性、高适应性和快速响应能力。

由于系统的庞大和复杂，被求解问题越综合、越形象、越直觉、越模糊，则用户和计算机系统之间的鸿沟就越宽，而实际上人们从主观愿望出发，十分迫切地想和计算机建立一个和谐的人机环境，使人们认识客观问题时的认识空间与计算机处理问题时的处理空间尽可能一致。为了适应 21 世纪信息的需求，人们不仅仅要求能通过打印输出或显示屏幕上的窗口，从外部去观察信息处理的结果，而且要求能通过人的视觉、听觉、触觉、嗅觉

以及形体、手势或口令，参与到信息处理的环境中去，从而取得亲身的感受。因而信息处理已不再建立在一个单维的数字化信息空间上，而是建立在一个多维化的信息空间中，建立在一个定性和定量相结合、感性认识和理性认识相结合的综合集成环境中。

(5) 各种支持系统的开发

在产品开发过程中，有许多环节的功能或作用能通过模型化或其它形式化的方法和手段表现出来，但在决策环节和其它活动，有着较多模糊的因素和机理，有待于科技发展的深入；还有开发活动中环境、组织等方面的随机和突发的事件，因而虚拟产品开发系统中必须有着各种支持系统来支撑。如产品设计支持系统、装配支持系统和测试支持系统等。各种支持系统的建立，有利于使系统从专业向领域方向拓延，成为更高层次的工具性系统，也有利于缩短用户和开发过程的产品的距离。

从以上所述的敏捷制造技术的发展及虚拟制造技术的发展可知，产品的协同设计是实现先进制造模式的最重要的一项技术，也是传统设计方法在新历史时期发展的必然趋势。

1.3 企业组织结构的变化

本章的上节介绍了 3 种先进制造技术，这 3 种先进制造技术重点从技术的角度给出了企业在激烈市场竞争环境下，为了赢得市场竞争从产品设计、企业组织上可采用的一些策略和设计制造技术。实际上先进制造技术的覆盖范围是相当广泛的，从企业经营过程的组织和流程的优化角度出发，先进制造技术还包括企业经营过程重组技术（本书 9.1 节对企业经营过程重组的概念有深入的介绍）、供应链管理、全面质量管理、准时制造、精良生产等，从保护环境的角度出发，先进制造技术还包括绿色制造、清洁化生产等，另外可称为先进制造技术的还有计算机集成制造、智能制造、全能制造等等。

并行工程、敏捷制造、企业经营过程重组、供应链管理等先进制造技术的一个共同的特点是强调过程集成和过程重组，这些特点也是先进制造技术明显的区别于传统制造技术单纯从技术角度提高设计水平、制造手段与生产率的做法。先进制造技术更强调管理技术、制造技术与信息技术的结合。而且由于先进制造战略的出现，使得企业的组织结构也在从传统的功能组织结构向过程组织结构变化。

传统的企业组织结构是功能部门制，即按照不同的功能和职能设立不同的部门，每个部门由若干人员组成，部门设立相应的部门负责人，每个下级部门从属于某个上级部门。上下级之间形成一个树型的组织结构。整个企业的结构呈如图 1.5 所示的金字塔式结构。

由图 1.5 上可以看出，企业的决策周期由上层到下层呈逐渐缩短趋势，不同层次的实时性由下向上呈逐渐降低的趋势。

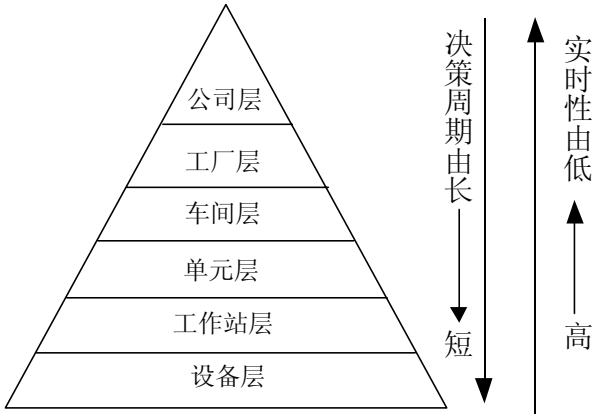


图 1.5 企业组织的金字塔结构

图 1.6 给出了一个示意性的企业树型组织结构。

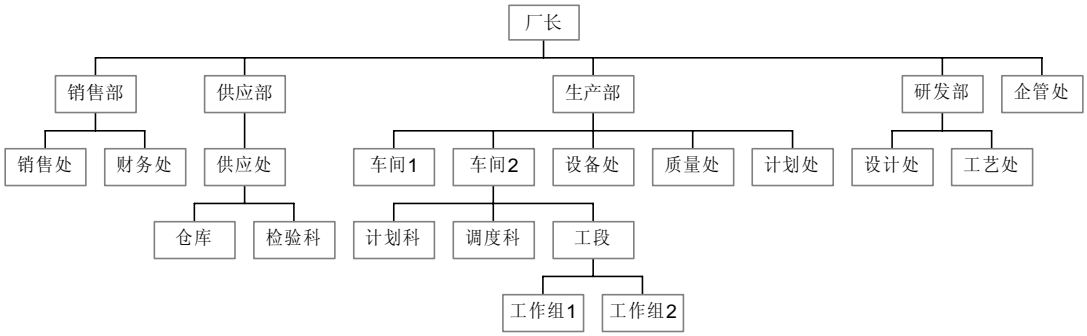


图 1.6 企业组织结构示意图

这种树型的层次组织结构已经明显的不能够适应当前日益竞争激烈的市场环境的要求，这种不适应主要反映在企业的柔性差、生产周期长、市场相应速度慢、客户需求满足度差。在传统的企业组织结构下，一个产品或者客户服务需要通过许多不同的功能部门，一个部门为许多不同的产品提供服务，为完成产品或者服务而执行的活动在不同的部门之间的传递逻辑复杂，因此，在未完成最终产品的制造前，客户几乎无法知道其定单的执行情况。图 1.7 给出了在层次组织结构下为完成客户定单进行的产品设计制造服务活动的传递流程。

在这种组织结构下，每个单元都由其上一级的功能单元进行管理，它的工作完成质量由上级进行评价，决策也由上级进行。因此，在出现问题的时候，每一级都会把责任推到上一级的功能单元，导致出现扯皮和责任不清现象，不利于解决问题和改进工作，这种职责不清现象最直接的后果是导致产品设计制造过程执行时间长和产品成本高。每个单元是对其上级负责而不是对用户负责，往往导致客户的满意度不高。另外，这种组织结构的柔

性非常低，在客户定单投入生产后，客户想改变定单的需求、功能或者性能要求是非常困难的，即使能够改变，其涉及的生产成本的增加也是非常大的。

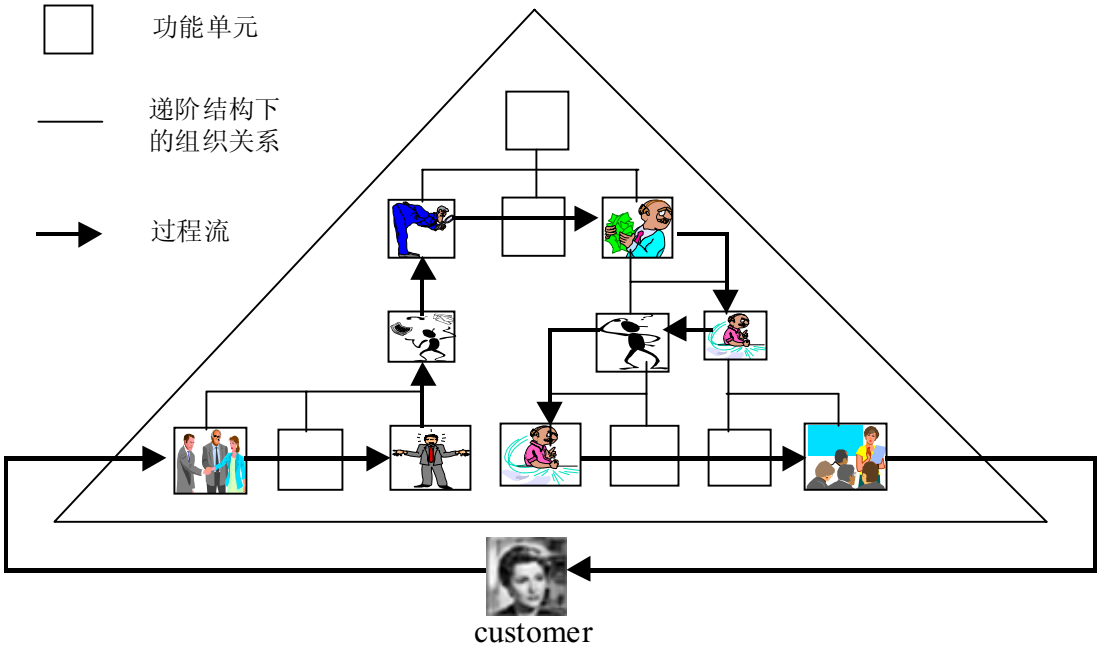


图 1.7 递阶组织结构下业务过程在不同功能单元之间的传递过程

然而产品客户化定制生产比例、客户希望了解产品设计生产流程情况的需求、企业希望通过改进业务流程以降低产品成本和生产周期的要求等内外因素的不断增强，迫使企业必须改变传统的组织结构，更好地适应日益变化的市场环境。这种变化的最终结果是将传统的功能组织结构改变成面向市场的过程组织结构。相对于传统的功能组织结构，过程组织结构有两个明显的特点，一是它们以外部或者内部的客户作为其过程的输出接收者，这样每个过程的执行结果都可以达到及时正确的检验，无论是内部还是外部的客户都可以对过程输出结果进行满意度评价；二是他们跨越传统的功能部门的边界，从而将原来割裂的业务过程集成起来，减少了不必要的部门间的协调过程和可能出现的扯皮现象。这种以过程组织结构给企业从根本上大幅度缩短市场响应时间、提高服务质量、产品质量提供了一种可行的方案。

因此，采用过程管理将使企业改变其传统的按照功能来配置其人员的组织结构，变成按照企业要实现的主要业务流程来配置组织结构，可以大大缩短其主要业务过程的处理时间，提高其对市场的响应能力。

图 1.8 给出了传统的面向功能的组织结构向面向过程的组织结构的转变。组织结构的改变将大大减少在企业内部不必要的物料、信息的传递时间。当然，整个企业组织结构的调整首先需要调整传统的以部门组织生产、人员从属于某个部门的做法，变成以项目来组

织生产和人员的工作方法。一个人可能同时从属于多个项目。

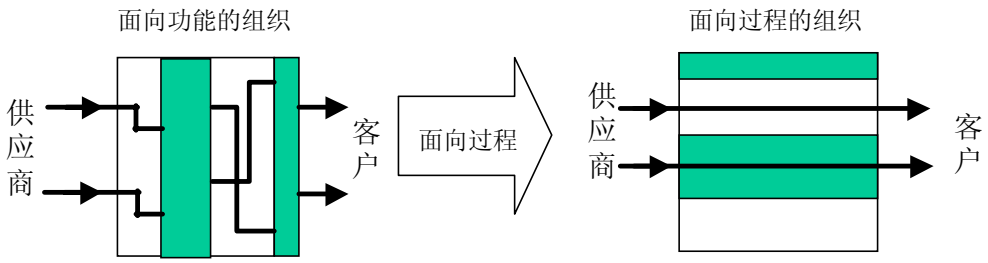


图 1.8 面向功能的组织到面向过程的组织的变迁

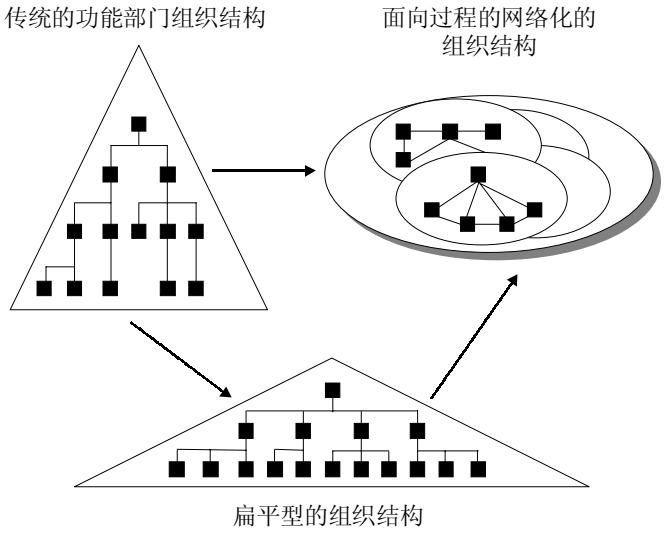


图 1.9 企业组织结构的变化趋势

企业组织结构的变化需要经历一个相当长的过程，除了需要企业领导提高意识和进行组织结构调整外，还需要在整个企业普及过程管理的概念和方法，并建立相应的管理制度和支撑环境。在当前环境下，为了提高企业对市场响应的灵活性和柔性，许多企业已经将传统的金字塔的瘦长型结构转变为扁平型的组织结构，这种扁平型的结构在尽可能的条件下减少企业的组织层次，减少不必要的决策环节，提高对市场的响应速度。这种扁平型的组织结构可以看成是由功能部门组织结构到面向过程的组织结构的过渡阶段，图 1.9 给出了由功能部门组织结构到面向过程组织结构的转化过程。图中箭头方向反映了企业组织结构的变化趋势。

那么，既然面向过程的管理可以给企业带来显著的效益，为什么当今绝大多数的企业还是采用传统的金字塔式的功能部门制的组织结构呢？除了企业领导者的意识外，一个重要的原因是面向过程的管理远远比功能部门制的管理过程要复杂，这种复杂性不仅表现在管理的环节数量多、过程跨度大（经常需要跨越不同的部门），而且表现在不同环节之间功

能进行协调的要求远远高于功能部门制的情况。以下我们对这种情况通过示例的方式进行简单的分析。

在传统的功能部门制结构下，按照最优管理的方法设计的管理模式，一个部门管理的下属机构通常不超过 7 个（根据人类认知科学和企业管理的经验，一个人直接管理的事务数量最好小于 7 个，才能够保证最高的管理效率），如果一个部门管理的下属机构超过 7 个，一般会将一部分机构进行重新分类并在他们之上定义一个新的中间机构，这个新的中间机构成为这个部门的下级并实现对从原来部门中分离出来的机构进行管理。如一个厂的车间超过 7 个，可能会在车间之上定义分厂这个机构，并使分厂管理的车间少于 7 个。因此，在功能部门制组织结构下，如果哪个管理部门由于管理的机构太多而导致效率低下，就会对组织结构进行调整以提高效率，而这种不断调整的结果就是产生多层的递阶组织结构，从表面上看，部门的效率提高了，但是相应的官僚主义和扯皮现象却大大增加了，实际上效率反而降低了。

上述的通过分解降低管理复杂性的方法在面向过程的组织下是行不通的。由于过程管理是面向市场、面向用户需求的，因此，它需要管理从用户定单生成、产品设计、制造、装配、运输，甚至用户最终付款的所有环节，这些环节又由许多小的环节组成。由于每个环节完成的功能不同，而且面向过程的管理需要对每个环节的完成情况进行跟踪，因此采用分解的方法来降低管理的复杂性的方法在面向过程的管理模式下行不通。也就是说，面向过程管理的本质是一种细致的全面管理模式，面向过程的管理将一个过程及其涉及的所有环节作为管理对象，无论对于简单的过程，还是复杂的过程，面向过程的管理的基本内容是几乎相同的。由此可以看出，面向过程管理的复杂性主要是被管理的对象（过程）的复杂性和面向过程管理的目标（全面、透明的、及时的过程环节管理）决定的。

另外，在面向过程的管理中，一个过程中的完成不同功能的环节之间的协调过程也远比功能部门制组织结构下要复杂，在功能部门制组织结构下，每个部门的首要任务是按时按量完成自己的工作，至于其他部门如何工作不需要他们考虑，即“各人自扫门前雪，不管他人瓦上霜”。如产品设计部门仅负责完成产品的设计，而对于工艺部门如何完成工艺、是否方便完成工艺设计、生产部门如何完成产品制造、产品制造是否成本最低等问题他们是不关心的，只有在出现工艺设计或者产品制造无法完成的情况，产品设计部门才会对产品的设计进行修改和改进。这种情况在许多企业中并不少见，由此导致产品生产周期长、成本高、质量低等问题也早已被许多工程技术人员和管理人员认识到。这种问题在面向过程的管理中可以得到较好的解决。过程管理不仅仅强调某个环节的完成情况，而是从整个

流程的角度来看待企业的业务经营过程，它把完成最终生产经营目标作为流程管理的目标。如以最终客户满意度作为评价某个流程执行情况的指标。所以，在面向过程的管理模式上，更强调不同环节之间的配合，强调总体流程的优化，它可以显著的提高效益，但是也显著的增加了管理上的复杂性和难度。

从上面的分析可以看出，使用人工管理的方式是不可能完成面向过程的管理工作，而且仅仅使用简单的事务处理软件（如财务软件、MRPII 软件、CAD 软件、车间调度软件）也不能够完成面向过程的管理。为了有效的处理面向过程管理模式带来的复杂性问题，必须采用能够支持过程建模、过程执行、过程集成、过程协调的软件。从目前现有的软件分类来看，在支持过程管理方面的软件主要有过程建模软件、过程仿真与企业经营过程分析工具、过程协调与群件软件、过程执行与过程自动化软件（主要是工作流执行与管理系统）。

过程建模软件的主要功能是完成过程的建模功能，这方面的软件有 IDEF3 软件、Petri 网软件、iGrafx 软件等；过程仿真与企业经营过程分析的软件完成对企业过程模型的静态与动态性能分析，发现企业业务流程中存在的瓶颈问题，计算企业流程运转的周期、成本、效率指标，对比不同的过程改进方案，为企业优化其经营过程提供决策的依据，这方面的软件工具有 ARIS 工具集、SimProcess、IThink、以及其他基于 Petri 网的过程仿真软件工具；过程协调与群件软件能够有效支持过程的协作管理、支持群组工作和群决策过程，它主要解决群组工作情况下，不同成员之间的通信、协作与协调问题，这方面比较著名的软件有 IBM 的 Lotus Notes，Microsoft 公司的 Exchange，Novell 公司的 GroupWise 等。工作流管理软件主要完成企业业务过程的自动化执行与监控，它将通过建模得到的企业业务过程模型进行实例化并投入运行，对运行的过程模型中的活动执行情况进行调度、管理和监控。工作流管理系统是企业实行过程管理的最重要的和最有力的支持工具。可以这样说，如果没有相应的工作流管理系统的支持，企业要实现面向过程的管理是非常困难的，对于业务过程较为复杂的企业，靠人工进行面向过程的管理几乎是不可能的。比较著名的工作流管理软件有 IBM 的 FlowMark，Staffware 公司 Staffware，Action 技术公司的 ActionFlow 等。

随着企业经营过程重组概念和方法在企业重视程度的提高，随着先进制造战略在企业的不断实施，随着企业的组织模式从面向功能的组织结构到面向过程的组织结构的逐步转变，作为支持过程建模、优化分析、经营过程自动化的有效支持工具，工作流管理技术与工作流管理系统软件在近年来得到了广泛的重视，其发展十分迅速，市场销售额以每年几乎 70%的速度递增，而且工作流管理技术已经在许多企业进行了成功的应用，取得了显著

的成果。

1.4 企业计算机应用的发展过程与趋势

随着计算机价格的持续下降、计算机功能的迅速增强、相关应用软件系统的日益丰富、网络技术和 Internet 技术的日益普及，计算机已经越来越成为人们工作和生活中不可缺少的工具。同样，计算机在企业的生产经营中也扮演着越来越重要的角色。

根据企业部门完成业务功能的不同，计算机在企业的应用方式和完成的工作也有一定的区别。我们将目前企业的计算机应用进行了一个简单的分类，如图 1.10 所示，整个大圆圈代表企业的计算机应用系统，其中有三大类应用可以覆盖企业计算机应用的绝大部分。这三大类应用是计算机管理信息系统、计算机辅助产品设计系统、计算机辅助生产控制系统。

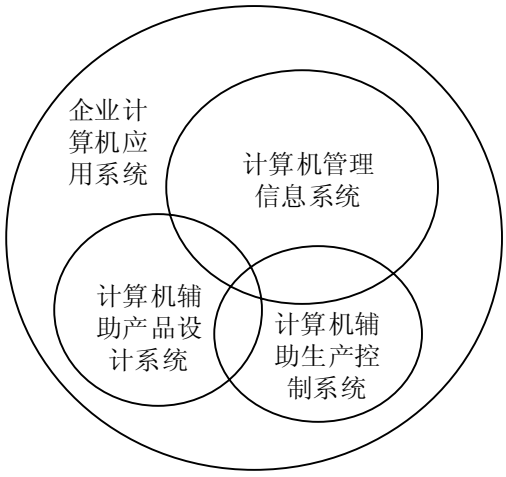


图 1.10 企业的计算机应用系统及其分类

计算机管理信息系统覆盖了企业的经营业务管理功能，如销售管理、财务管理、计划管理、库存管理、采购管理、办公自动化、人事管理、运输管理等等，计算机管理信息系统包含了企业约 80%的信息量，它是企业管理现代化的最主要的管理工具。计算机辅助产品设计系统是企业实现产品研制现代化或设计自动化的主要支持系统，它包括计算机辅助设计系统（CAD）、计算机辅助工艺系统（CAPP）、计算机辅助制造系统（CAM）、计算机辅助工程（CAE）、计算机辅助夹具系统（CAF）、产品数据管理系统（PDM）等等，计算机辅助产品设计系统是企业提高产品设计水平、提高产品质量、降低产品成本、缩短产品研制周期、提高企业产品创新能力的重要工具。计算机辅助生产控制系统是提高企业产品制造水平、产品质量、生产率水平的重要工具。在离散型工业，计算机辅助生产控制系统的功能包括车间生产设备管理、生产计划制定、作业调度、数控机床的计算机控制、自动

运输设备的控制、自动化仓库的管理、在制品管理、物料管理等等，其主要目标是提高产品质量、缩短产品生产周期、减少在制品的等待时间、提高关键设备的利用率等；在连续型工业，如化工过程、炼油工业、钢铁工业、煤炭工业中，计算机辅助生产控制系统主要是采用集散控制系统（DCS）、可编程控制器（PLC）、现场控制设备等控制与监测设备实现对于连续高速生产线的控制，其目标是提高产品质量、提高生产率、尤其是减少因为故障造成的停产现象。

虽然在图 1.10 中给出了企业计算机应用的一种分类方法，但是在实际企业应用中，有一部分系统是难以将其明确的划分为某一类，比如设备管理软件，它即可以属于管理信息系统也可以属于生产控制系统。又比如质量管理系统，它的一部分功能可能属于管理信息系统，而另外一部分功能可能属于生产控制系统。所以，图 1.10 给出的分类仅是一个参考分类方法，读者在实际应用中应根据具体情况进行合理的划分。

除了上面介绍的企业计算机应用的划分方法外，还可以根据计算机在企业的应用发展历史对企业的计算机应用情况进行阶段划分。在此我们给出一种根据企业计算机应用软件集成度和集成范围来划分企业计算机应用的分类方法。根据这种分类方法，我们可以将企业的计算机应用划分为五个阶段。图 1.11 给出了企业计算机应用的五个发展阶段的图示化表示形式。以下对企业计算机应用发展的五个阶段进行简要的介绍：

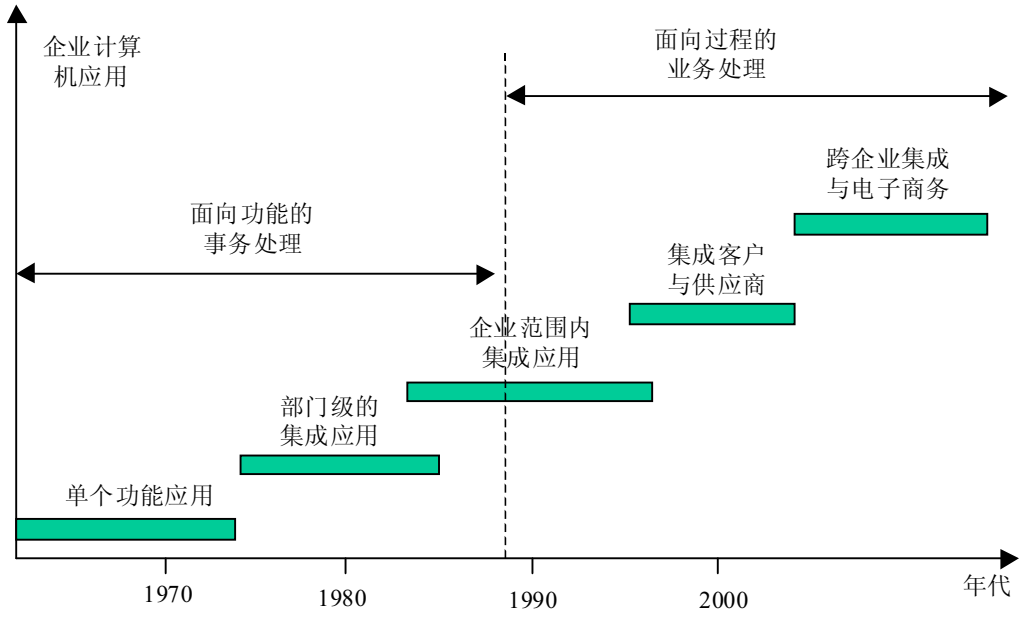


图 1.11 企业计算机应用的五个发展阶段

(1) **单个功能应用：**20 世纪 70 年代中期以前，企业的计算机应用水平还较低，所开发的计算机应用功能也比较简单，主要解决局部的计算功能，如财务记帐、生产计划制定、采购物料数量品种计算、库存统计、计算机辅助绘图等等。

(2) **部门级的集成应用：**20 世纪 70 年代中期至 80 年代中期，开始应用具有一定集成度的综合应用软件，如 MRPII 系统实现了财务、库存、采购、计划的集成，CAD 软件实现了产品造型与绘图的功能，这些集成的应用将企业的计算机应用水平推上了一个新高度。

(3) **企业范围内集成应用：**20 世纪 80 年代中期开始的以计算机集成制造技术（CIM）、企业资源规划（ERP）、产品数据管理（PDM）为代表的企业范围内的集成应用。ERP 扩展了 MRPII 应用的范围，它不仅包含了 MRPII 的基本功能，还包括如客户关系管理、售后服务、项目管理、集成化的过程管理等功能。在产品研制和设计自动化应用方面，在 PDM 系统的基础上，对产品研制过程和研制过程中涉及的产品数据进行集成化的管理，并实现了 CAD/CAPP/CAM 集成的计算机应用，这些工作促进了产品的设计自动化水平。而 CIM 则是先进制造技术的典型，它从整个企业信息集成与系统优化的角度出发，强调信息集成和企业计算机应用的整体规划，以期提高整个企业的 T、Q、C、S 水平。CIM 技术的成功应用大大增强了企业采用先进的信息技术改造和提升传统制造业的信心，加快了企业计算机应用的步伐。无论是 ERP、PDM 还是 CIM，它们都已经从支持功能事务处理的计算机应用技术发展到支持过程集成与过程管理的业务处理的计算机应用阶段。

(4) **集成客户与供应商：**自 20 世纪 90 年代中以来，供应链管理（supply chain management）技术得到了企业广泛的重视。供应链管理的基本观点是在整个供应链中，每个企业（无论是原材料供应商、部件供应商、整机装配厂、还是代理商和销售商）都是为最终客户提供服务的整个服务链中的一个环节，绝大多数的厂家既是为后续厂商提供产品的供应商，又是前趋厂家提供产品的客户。因此，快速高效的实现供应链中相关环节之间的信息交换，降低信息交换的事务成本有利于为最终客户提供低成本高质量的服务，并且可以对市场的变化进行快速的响应。供应链管理软件通过与企业的 ERP 或者相应管理软件的集成，并通过电子数据交换、工作流管理软件、WEB 等方式迅速交换信息来达到上述目标。通过供应链管理，企业可以与客户和供应商建立紧密的联系，并且规范供应链中的上下游的关系。在供应链管理中，不仅要组织好企业的内部信息，更重要的是要实现整个供应链上下游之间的过程管理。

(5) **跨企业集成与电子商务：**在本章介绍敏捷制造战略时，我们已经指出，敏捷制造已经成为 21 世纪制造业将采用的主要先进制造战略，动态联盟和虚拟企业是实施敏捷制造战略的主要实施方式。实施动态联盟必须要实现结盟企业之间的信息集成，并在此基

础上实现过程集成。前三个企业计算机应用阶段主要是企业内部信息集成，从供应链管理技术的应用开始，企业的计算机应用已经进入跨企业（或者称为企业间）的信息集成。作为跨企业间信息集成的重要手段之一的电子商务技术近来得到了广泛的重视，许多大型企业和集团纷纷推出自己的电子商务系统，希望通过电子商务这种低价位的新型方式来获得更大的市场份额。无论是跨企业集成，还是电子商务目前还都处于发展阶段，涉及跨企业集成的许多使能（enabling）技术，如跨企业的分布式工作流管理技术、计算机支持的协同工作（CSCW）技术、分布式智能代理技术，都还处于关键技术攻关和初步应用阶段。

从图 1.11 中，我们还对企业计算机的应用阶段按照面向功能的事务处理系统和面向过程的业务处理系统二种方式进行了划分。在 20 世纪 80 年代后期以前的企业计算机应用主要是面向功能的事务处理系统，其开发实施的目的主要是提高企业事务处理的效率、降低事务处理成本。20 世纪 80 年代末开始发展的企业计算机应用逐步加强了对过程管理的重视，并强化了面向过程管理的业务处理系统功能，如在 ERP 软件中增加了项目管理、工作流管理功能，在 PDM 软件中增加流程管理功能。实施面向过程管理系统的目的主要是从提高整个经营过程效益、提高客户满意度、提高经营过程柔性等影响企业市场竞争力的因素入手，按照经营业务而不是按照功能，根据最终经营目标而不是局部优化的思路，在面向市场、面向客户的指导方针下发展企业的计算机应用系统。

从先进制造战略、企业经营过程重组、市场环境的变化、到企业组织结构的变化趋势、以及企业计算机应用的发展历史可以清楚地看出，面向过程的计算机应用在今后的企业经营业务中将发挥越来越重要的作用。作为面向过程建模、优化、执行与监控的工作流管理技术已经表现出其强大的生命力，其广阔的市场前景和应用价值已经受到了越来越多计算机软件供应商和研究机构的重视。可以预计，在今后的几年内，工作流管理技术将向 MRPII、ERP、PDM 技术一样，掀起企业计算机应用的新高潮。

第 2 章 工作流管理系统基本概念

2.1 工作流问题的起源与基本概念

2.1.1 工作流问题的起源

工作流是从英文单词 Workflow 翻译得来的。Workflow 是由单词 Work 和 Flow 组合得到的词。单词 Work 表示工作或者任务，单词 Flow 的含义是流动、流程或者流量等。Flow 反映了一种变化以及变化的过程，它本身的含义比较抽象，但是当它与某个具体的过程相联系时就有了具体的含义，如电流、水流、气流。在企业的经营管理与生产组织中，Flow 也有重要的意义，如表示物料传输过程的物料流、表示资金流动的资金流、反映信息处理和传递过程的信息流，同样还有价值流、决策流、控制流等概念。依此类推，用活动及活动之间变化的过程表示的业务流程就是工作流。

工作流的概念起源于生产组织和办公自动化领域。它是针对日常工作中具有固定程序的活动而提出的一个概念。提出的目的是通过将工作分解成定义良好的任务、角色，按照一定的规则和过程来执行这些任务并对它们进行监控，达到提高办事效率、降低生产成本、提高企业生产经营管理水平和企业竞争力的目标。实际上，自从进入工业化时代以来，有关过程的组织管理与流程的优化工作就一直在进行，它是企业管理的主要研究内容之一。只不过在没有引入计算机信息系统的支持以前，这些工作是由人工来完成的。在计算机网络技术和分布式数据库技术迅速发展、多机协同工作技术日臻成熟的基础上于 20 世纪 80 年代中期发展起来工作流技术为企业更好地实现这些经营目标提供了先进的手段。工作流技术一出现马上就得到广泛的重视和研究。至今工作流管理技术已成功地运用到图书馆、医院、保险公司、银行等行业，然而它更重要的应用还是在工业领域，特别是制造业领域中。

在企业应用实际中，虽然工作流的概念相对于物料流、资金流、信息流等概念要抽象一些，但是工作流从更高的层次上提供了实现物料流、资金流、信息流及其涉及的相关过程与应用的集成机制，从而使得企业能够实现业务过程集成、业务过程自动化（不是物流自动化和信息处理自动化）与业务过程的管理。在工作流概念下实现业务过程集成与业务过程自动化的集成机制是通过定义不同任务之间相互关系的工作流模型（也称为过程模型）来实现的。在工作流模型中，无论是具体的物料转移动作、实际物理装置的操作动作、还

是抽象的信息处理动作与决策过程，都可以用工作流的基本组成元素－－任务（也称为活动）来统一地进行描述。同样，反映不同任务之间的关系，无论是具体的车间中零件加工顺序关系、办公自动化中的文件批转、还是抽象的决策流之间的关系都可以用工作流的基本组成元素－－连接弧来统一地进行描述。连接弧反映了对企业业务经营过程的一种控制逻辑，它定义了活动之间的连接关系和执行顺序。

传统的企业计算机管理信息系统的主要功能有三个，即信息处理、事务处理与决策支持。信息传递和信息处理构成了企业（包括制造企业、商业企业、服务企业）和行政管理部门的业务工作中的主要内容之一，也是计算机管理信息系统的主要功能之一，它是企业进行事务处理（如销售定单处理、经营计划制定、物料采购计划生成、车间作业分配、库存管理等）和经营决策的基础。在信息传递和信息处理结果的基础上，各级领导进行相应的决策活动，这些决策活动决定了企业业务的开展方式和经营战略。如何方便地在不同的业务部门、业务人员之间高效地进行信息传递是企业领导、业务人员、包括现在的计算机软件开发人员十分关心的问题。由于信息需要一定的载体和方法才能够实现有效的传递，在计算机软件没有成为主要的业务支持工具前，实现信息传递的最好方式是通过纸张作为载体，利用通知、文件、信函、传真、报告等方式在不同的业务部门、业务人员之间进行。

这种以传统的纸张为载体的信息传递与处理方式的效率很低，需要花费相当的人力、物力来完成信息的处理、组织、存储以及查询检索，同时这种方式降低了对客户需求的响应速度，给企业的生产经营都带来不利的影响。在计算机得到了广泛普及和企业的计算机应用水平日益提高的情况下，企业业务人员希望能够以一种无纸化的、计算机使能的工作环境来开展其日常的业务工作。一些公司和企业因此建立了自己专用的或者可商品化的表单传递应用系统（Forms-routing applications）用来实现日常表单处理的电子化与自动化。这些系统通常是以主机－－终端方式运行在大型机或小型机上，用户（业务人员）通过终端运行位于主机上的应用程序，它们可以看成是现在工作流管理系统的雏型，只不过它所适用的环境还比较简单，所提供的功能还不全面，性能与系统的结构也不够先进。

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、复合文档、结构化路由（structured routing）、实例跟踪、关键字索引以及光盘存储等功能结合在一起，形成了一种全面支持某些业务流程的集成化的软件（包），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 商用系统，ViewStar 于 1988 年推出的 ViewStar，IBM 于 1988 年推出的 ImagePlus。它们的出现使许多企业很快认识到在业务流程的处理过程中，纸张有可能只出

现一次，即承载文档的纸张在流程的起始处将首先被扫描成电子文档，继而后续对文档的处理过程也将全部实现电子化。如果流程的输入本身就是电子化文档，那么自始至终在系统中所传递的将完全是存储在计算机磁盘中的数据。很显然，这种集成化软件系统为企业简化与重组自己的关键业务流程提供了一种非常好的方法。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 20 世纪 90 年代，随着计算机与网络技术的迅速发展，特别是在 Internet 应用日益普及的情况下，现代企业的信息系统的分布性、异构性和自治性的特征越来越显著，相应的企业信息资源也分布在异构的计算机环境中，信息源之间的连接表现出松散耦合的特点，这样的信息系统环境简称 HAD 环境（异构、自治、分布）。企业物理位置的分散性和决策制定过程的分散性特征日益明显、对日常业务活动详细信息的需求日益提高、Client/Server 体系结构和分布式处理技术（CORBA、WWW、OLE、JAVA）的广泛应用，以上这些情况都说明了这样一个事实——集中式信息处理的时代即将成为过去，取而代之的将是大规模的异构分布式信息处理与应用执行环境。在这种大规模的分布式环境下高效的运转相互关联的任务，并且对执行的任务进行密切监控已成为一种发展趋势。在这种技术背景下，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。

目前，在全球范围内，对工作流的技术研究以及相关的产品开发了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、Internet 服务等都已被容纳到工作流管理系统之中。工作流产品的市场每年以两位数字的速度迅猛增长。市场上工作流产品发展迅速，据统计，1997 年市场上约有 70 多种工作流产品在相互竞争， 97 年工作流产品的市场增长率超过 35%。而且随着计算机技术的发展，工作流产品的供应商又及时地将新的技术融入工作流中，提高产品性能，使得工作流技术得到不断完善。作为支持企业经营过程重组（Business Process Reengineering－BPR）、经营过程自动化（Business Process Automation－BPA）的一种手段，工作流技术的研究应用日益受到学术界与企业界的重视。许多大学和研究机构也致力于工作流技术的进一步发展，开展了一系列研究项目，取得了显著的成果。

2.1.2 工作流的基本概念和定义

1993 年工作流管理联盟（Workflow Management Coalition, WfMC）的成立标志着工作流技术开始进入相对成熟的阶段。为了实现不同工作流产品之间的互操作，WfMC 在工作流管理系统的相关术语、体系结构及应用编程接口（WAPI）等方面制定了一系列标准。

不同的研究者和工作流产品供应商从不同的角度给出了工作流的定义。以下我们给出几个具有代表性的定义，供读者参考。

工作流管理联盟给出的工作流定义是[1]：***工作流是一类能够完全或者部分自动执行的经营过程，它根据一系列过程规则，文档、信息或任务能够在不同的执行者之间进行传递与执行。***

Georgakopoulos[3]给出的工作流定义是：***工作流是将一组任务（Task）组织起来完成某个经营过程。在工作流中定义了任务的触发顺序和触发条件。每个任务可以由一个或多个软件系统完成，也可以由一个或一组人完成，还可以是由一个或多个人与软件系统协作完成。任务的触发顺序和触发条件用来定义并实现任务的触发、任务的同步和信息流（数据流）的传递。***

还有许多关于工作流的不同定义，如 PeopleSoft 公司给出的定义是：***工作流是一个用来实施经营过程实践的机制***[4]。IBM Almaden 研究中心给出的工作流定义是[5]：***工作流是经营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步骤的定义、步骤间的执行顺序、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。***

以上的这些工作流的定义，包括其它的一些工作流定义基本上都是用非形式化语言对工作流所进行的描述，虽然表述方式略有不同，但是基本上都说明这样一个问题，即工作流是经营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。使用工作流来作为经营过程的实现技术首先要求工作流系统能够反映经营过程的如下几个方面的问题：即经营过程是什么（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，也就是控制流与信息流的定义）、由谁来做（人或者计算机应用程序，也就是组织角色的定义）、做得怎样（通过工作流管理系统对执行过程进行监控）。

根据以上定义和我们对工作流管理技术的理解，我们给出如下的工作流定义：

“工作流是一种反映业务流程的计算机化的模型，它是为了在先进计算机环境支持下

实现经营过程集成与经营过程自动化而建立的可由工作流管理系统执行的业务模型。”

在我们给出的定义中强调工作流模型是可被工作流管理系统执行的，这主要是为了区分工作流模型和一般意义上的过程模型。通常描述一组活动及其他们之间相互连接关系的模型可以通称为过程模型，但是并不要求这些过程模型用计算机来进行执行，如描述项目实施计划的 GANTT 图就是一个过程模型，但是一般并不需要用计算机来执行这个模型。而工作流模型从建立的目的就是为了实现业务过程自动化，要有计算机来进行执行。这就要求工作流模型不仅能够描述活动及其他们之间相互连接关系，而且需要定义许多其他的信息，如组织、资源、数据等，这样才能够由计算机进行解释和执行。另外一方面，由于工作流模型需要由计算机来执行，这就对工作流模型的准确性提出了更高的要求，工作流模型的定义也更加严格、准确。

以上的一些定义从不同角度说明了工作流是具有广泛应用价值的计算机软件技术，它更多的与经营过程发生关联，可以应用于经营过程的不同阶段。图 2.1 给出了一个称为工作流伞的示意图，反映了工作流覆盖的经营过程的范围与对应的工作流研究领域。

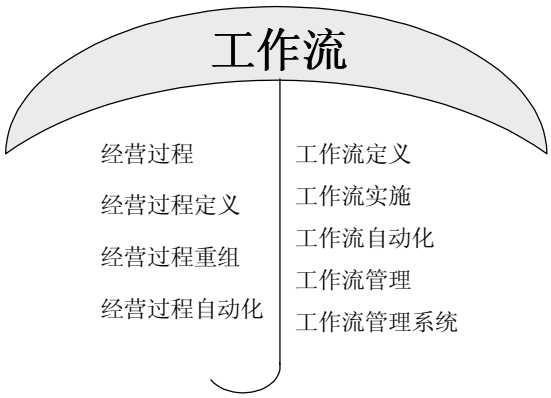


图 2.1 工作流伞

在实际情况中可以更广泛地把凡是由计算机软件系统（工作流管理系统）控制其执行的过程都称为工作流。工作流通常用于过程的自动化，通过将文档、信息或任务按照预先定义好的规则和流程在参与者之间进行传递，从而帮助用户实现或完成整个经营目标。在企业应用中，工作流经常与经营过程重组相联系，它完成对一个组织（或机构）中核心经营过程（或者称为关键经营过程）的建模、评价分析和操作的实施。虽然并非所有的 BPR 过程都需要采用工作流的方式进行实施，但是工作流技术通常是实施 BPR 的一个较好的方法，因为工作流提供了经营过程逻辑与它的信息支撑系统的分离，并实现了应用逻辑和过程逻辑分离，这种方式在进行企业实际应用时具有显著的优点。它可以在不修改具体功能模块实现方式（硬件环境、操作系统、数据库系统、编程语言、应用开发工具、用户界面）

的情况下，通过修改（重新定义）过程模型来改进系统性能，实现对生产经营过程部分或全部地集成管理，有效地把人、信息和应用工具合理地组织在一起，提高软件的重用率，发挥系统的最大效能。工作流技术可以支持企业实现对经营管理和生产组织的过程控制以及决策支持，它能够实现现代企业对“在适当的时间把适当的信息传给适当的人”的要求。工作流系统还可以提供系统日志功能，这种日志记录对于进行企业经营过程的运转情况的事后分析和流程优化提供了十分重要的数据。

如前面介绍，工作流主要是用来描述经营过程的，因此一个工作流就可以看成是企业的一个具体的经营流程的抽象或图示化的表示。那么什么是经营过程呢？我们知道企业的经营过程是由一系列相关任务组成，这些任务按照企业的管理规章和业务流程顺序或并行执行，最终完成企业的经营目标，如提供一种产品和服务。用一个比较规范的描述，我们给出经营过程的如下定义：

“一个经营过程是为了实现企业某个经营目标的一个过程，它在部分或者全部组织机构和人员的参与下，利用企业资源（包括所需的处理设备、通信设备、计算机硬件、软件等等），按照预先确定的规则，在参与者和组织机构之间进行文档、信息、任务的传递和处理（包括经营决策），从而实现预定的经营目标”。

那么如何才能够描述清楚一个企业的经营过程呢？首先我们认为要描述一个企业的经营过程，主要应该说明以下几个问题：

1) 这个经营过程要做什么？即其目的或想达到的目标是什么？

2) 这个经营过程是如何完成的，有那些任务并经过那些步骤完成？

3) 这个经营过程有谁参与完成，有那些部门参与？

4) 这个经营过程用了那些方式或手段来完成？

为了能够说明以上四个方面的问题，并且以计算机可以识别的方式建立企业经营过程模型，在工作流中定义一系列的基本概念和术语用来描述模型的组成，从而实现对企业经营过程的建模。首先是工作流的定义，如我们在前面介绍的，工作流就是将一组任务组织起来完成某个经营过程。所以工作流整个模型就是为了说明经营过程的目的，或者说这个模型描述的经营过程的目标。工作流中两个最基本的元素是活动和活动之间的连接关系。活动对应于经营过程中的任务，主要是反映经营过程中的执行动作或操作。活动之间的连接关系代表了经营过程的规则和业务流程。一个工作流就是一个用一组连接关系组合起来的一组活动组成的一个反映企业业务过程的模型。执行活动和活动之间的连接关系说明了如何完成企业的经营过程，包括完成经营过程需要完成那些任务和采用的步骤。

当然描述一个企业的业务过程不是仅有活动和活动之间的连接关系就能够描述清楚的。一个企业的经营过程中还要涉及参与操作的人员、组织、所操作的数据、使用了那些计算机应用程序等。在工作流模型中通过定义活动的角色（操作人员）和组织单元（组织结构、部门）来描述企业的经营过程是由谁来完成的。另外，通过定义工作流应用程序来说明采用了什么手段来完成经营过程。

有关工作流模型的各个组成元素及其具体含义，在本书的第 3 章有详细的介绍。下面用二个简单的例子来说明可以采用工作流建模方法进行描述的经营过程。

例 1: 某电脑公司计算机销售过程的工作流描述方法。图 2.2 给出流程的具体含义为用户通过 Internet 向公司发出定单，用户在填写定单时提出计算机的基本配置要求，并指定所需计算机的台数。整个流程通过以下活动完成销售业务过程。

1．用户通过 Internet 或其它方式向公司发出定单，指定所需的计算机基本配置要求和数量；

2．公司收到用户定单；

3．公司对用户定单进行检查（以下 3 个活动属于并发活动，同时进行）：

1） 计算价格；

2） 检查零部件库存是否满足需求；

3） 进行配置检查，确认用户的定单技术上可行；

4．进行决策；

1） 如果通过检查，则继续进行；

2） 如果未通过检查，则向用户发出信笺，解释为什么定单不能完成，并提出修改意见；

5．准备接收定单的确认通知，并要求用户付款；

6．发出通知；

7．装配计算机；

8．送货。

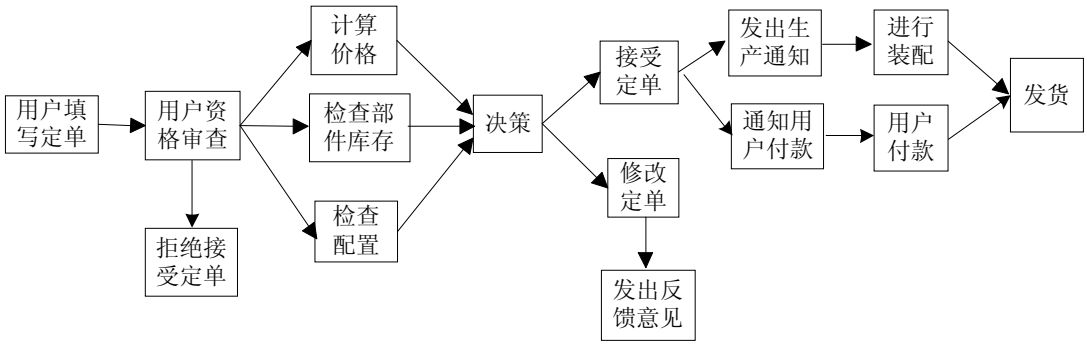


图 2.2 计算机销售过程的工作流程

例 2. 客户到银行取款的处理过程。图 2.3 给出的银行客户取款的具体过程如下：

1．客户填写取款单；

2．银行出纳接收取款单和客户的存折；

3．银行出纳检查客户存款信息，

1） 如果余额不足，要求客户重新填写取款单，返回步骤 1；

2） 存款余额足够，则继续进行；

4．客户输入密码；

1） 如果密码正确，继续进行；

2） 如果密码有误，重新输入，如果连续三次输入错误，则退出；

5．出纳取出相应的现金，并在客户的存折上进行记录；

6．将存折和现金交给客户。

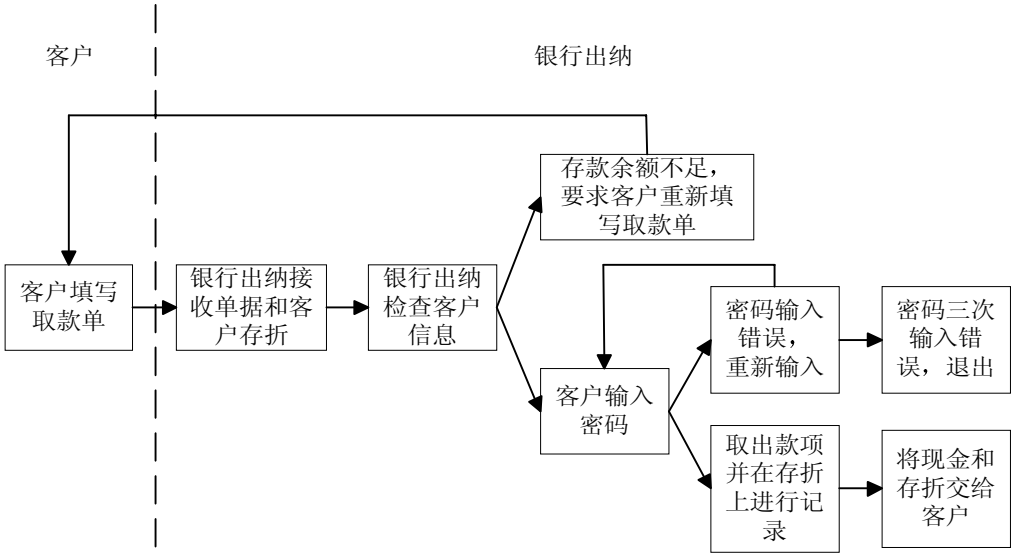


图 2.3 银行客户的取款过程

上面两个例子中的方框表示经营过程中的任务，它们对应于工作流中的活动。方框之间的连接弧表示了活动之间的关联。这二个例子表明了工作流技术具有广泛的应用背景，它可以用直观的、用户非常容易理解的方式来描述日常的事务处理活动和企业的经营过程。读者可以按照例子中介绍的描述方式来描述自己在工作中和日常事务中经常处理的流程，从而加深对于工作流技术的理解。如大学生可以描述每学期的学籍注册过程，软件工程师可以用工作流来描述软件的设计开发流程，机械设计师可以用工作流来描述产品的设计开发过程，行政管理人员可以用工作流来描述会议的组织过程和文件的批阅过程。

2.2 工作流管理系统

在明确了工作流基本概念的基础上，下面介绍工作流管理系统的定义。WfMC 给出的关于工作流管理系统的定义是：***工作流管理系统是一个软件系统，它完成工作流的定义和管理，并按照在计算机中预先定义好的工作流逻辑推进工作流实例的执行。***

我们给出的工作流管理系统的定义是：

“工作流管理系统是支持企业经营过程高效执行并监控其执行过程的计算机软件系统。”

通常，工作流管理系统指运行在一个或多个称为工作流机的软件上的用于定义、实现和管理工作流运行的一套软件系统，它和工作流执行者（人、应用）交互，推进工作流实例的执行，并监控工作流的运行状态。在这里需要强调指出的是工作流管理系统不是企业的业务系统。在很大程度上，工作流管理系统为企业的业务系统运行提供一个软件支撑环境，非常类似于在单个计算机上的操作系统。只不过工作流管理系统支撑的范围比较大、环境比较复杂而已，所以也有人称工作流管理系统是业务操作系统（BOS—Business Operating System)。在工作流管理系统的支撑下，通过集成具体的业务应用软件和操作人员的界面操作，才能够良好地完成对企业经营过程运行的支持。所以，工作流管理系统在一个企业或部门的经营过程中的应用过程是一个业务应用软件系统的集成与实施过程。

工作流管理系统可以用来定义与执行不同覆盖范围（单个工作者、部门、全企业、企业间)、不同时间跨度（分钟、小时、天、月）的经营过程。这完全取决于实际应用背景的需求。按照经营过程以及组成活动的复杂程度的不同，工作流管理系统可以采取许多种实施方式，在不同的实施方式中，所应用的信息技术、通信技术和支撑系统结构会有很大的差别。工作流管理系统的实际运行环境可以是在一个工作组内部或者在全企业的所有业务部门。

虽然不同的工作流管理系统具有不同的应用范围和不同的实施方式，它们还是具有许多共同的特性。从比较高的层次上来抽象地考察工作流管理系统，可以发现所有的工作流管理系统都是提供了 3 种功能；

1) 建立阶段功能：主要考虑工作流过程和相关活动的定义和建模功能；
2) 运行阶段的控制功能：在一定的运行环境下，执行工作流过程，并完成每个过程中活动的排序和调度功能；
3) 运行阶段的人机交互功能：实现各种活动执行过程中用户与 IT 应用工具之间的交互。

图 2.4 给出了工作流管理系统三个主要功能之间的关系。根据图 2.4 给出的工作流管理系统的主要特性，下面分别介绍它所提供的 3 个主要功能所涉及的研究工作和实施技术。

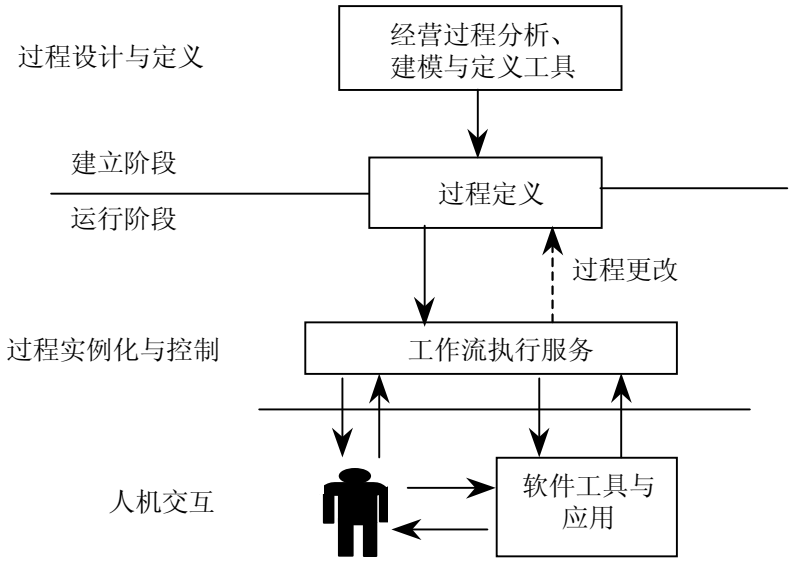


图 2.4 工作流管理系统的特性

2.2.1 过程建模

工作流管理建立阶段的功能主要完成经营过程的计算机化的定义。在这个阶段，利用一个或多个建模技术与工具，完成实际的经营过程到计算机可处理的形式化定义的转化。所得到的定义通常可称为过程模型、过程模板、过程元数据或过程定义。因此，在工作流建立阶段主要完成过程建模工作。在 WfMC 定义的工作流管理系统中，将过程建模得到的结果统称为过程定义。

过程建模是经营过程分析与经营过程重组的重要基础。过程建模主要解决如何根据过程目标和系统约束条件，将系统内的活动组织为适当的经营过程的问题。过程建模的作用体现为：

1) 用于准确描述企业的经营过程，供流程分析和优化（如经营过程重组）使用。

2) 用于在不同的组织和信息系统间共享经营过程知识，便于实现基准工程（Benchmarking），以及企业动态联盟。

3) 用于企业 CIMS 实施，根据设计的企业过程模型进行相应的功能构件配置，使得所建立的系统能够按过程实现横向集成，而不是按传统的部门划分结构实现纵向集成，从而满足企业核心价值流的要求。按过程模型进行系统构件配置还能够实现柔性更好的过

程集成。

4) 用于研究、开发新的经营过程，以满足不同业务需求和企业动态结构演化。

有很多方法可以用来进行工作流（过程）模型的定义与描述。过程建模方法学研究的主要内容和目的是为企业经营过程建模提供一套完整有效的描述经营过程的建模语言。对流程的描述需要提供对逻辑顺序结构，如顺序、分支、汇合、条件、循环、并行的描述。使用者可以通过这一套语言来对企业的经营过程进行形式化描述。目前较为广泛接受的建模语言有 CIMOSA[6]的经营过程描述语言、工作流管理联盟 WfMC 定义的工作流描述语言、Keller[7]等人提出的 EPCM 模型等，这些工作流描述语言的描述形式与程序设计语言中语义结构的定义方式类似。其它一些方法是采用传统项目管理中使用的概念和模型来表述经营过程，例如 PERT 图或其它各种形式的网络图等，这方面的工作可以参考文献[7]。后一类方法易于在已有的项目管理软件工具的基础上扩展得到，所以在实用系统中采用得也比较普遍。当所需建模分析的经营过程比较复杂，并存在并发、冲突等情形时，采用 PERT 图或网络图等简单的描述方法就很难将问题描述清楚，这时就需要采用形式化程度更高、描述能力更强的方法，如 PETRI 网方法。

本书第 6 章给出了过程建模方法的详细介绍。

2.2.2 工作流运行控制

在完成了过程模型的定义后，所生成的工作流模型将由工作流执行服务软件进行实例创建并控制其执行过程。工作流执行服务对使用工作流模型描述的过程进行初始化、调度和监控过程中每个活动的执行，在需要人工介入的场合完成计算机应用软件与操作人员的交互。这样，工作流执行服务实现了在模型中定义的经营过程与现实世界中实际过程之间的连接。这个连接通过工作流执行服务与应用软件、操作人员的交互来完成。实现这个连接的核心功能是工作流管理软件，工作流管理软件又称为工作流机。

工作流机除了完成过程的创建、删除、活动的执行与控制外，它的另外一个重要的功能是完成与应用软件及操作人员的交互。这是因为在实际企业应用中，应用软件和操作人员是完成经营业务工作的主体，而工作流机通过过程定义和活动之间顺序控制实现这些独立的功能实体间的集成，从而使整个企业经营活动成为一个协调运行的整体。

企业经营过程的执行通常需要若干个应用软件和若干人员的参与才能够完成，但是任何一个企业的实际应用都是在具有分布性和异构性的计算机网络环境中运行。分布性是指

应用软件运行在不同地点的不同计算机系统上，异构性是指应用软件运行在不同的计算机硬件环境、操作系统、数据库管理系统上。以机床制造厂为例，它的定单管理系统可能是运行在 PC 机、Windows95 操作系统和 SQL Server 数据库上的利用 Powerbuilder 开发工具开发的应用软件,而它的产品设计系统可能是运行在HP工作站、HP—Unix操作系统和Oracle数据库上的大型 CAD 软件（如 Pro/E、I—Deas），它的生产计划和物料计划系统可能是运行在另外一个由小型机和 PC 机组成的客户/服务器结构下的计算机软件系统上，这就是一个典型的分布异构计算机系统。为了能够支持这样的企业的生产经营过程，作为工作流管理控制软件的工作流机同样需要在分布异构的环境中运行。

对工作流机的分布性要求客观上是由企业的实际运行环境决定的，工作流管理系统可以采用不同的方法来满足企业应用对于分布性的要求。按照工作流机管理系统设计开发的难易程度，工作流管理系统的分布性可以分为分布式的工作流用户与应用接口、分布式工作流机和分布式工作流模型三种主要的分布方式。分布式的工作流用户与应用接口通常是工作流管理系统必须提供的分布处理功能，因为企业的应用软件和用户本身是分布在不同的计算机环境和不同的工作地点。

图 2.5 给出了一种分布式的工作流执行服务情况。

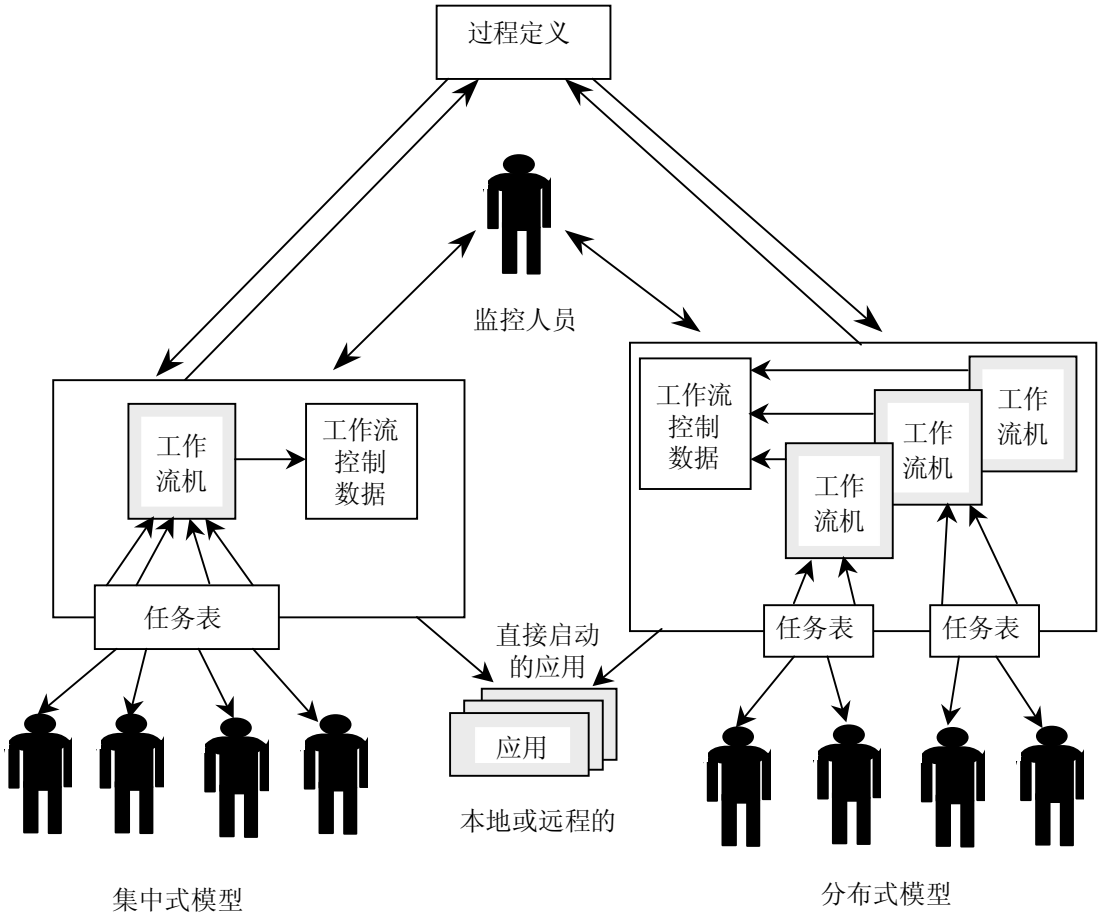


图 2.5 分布工作流机与应用结构

其中左面表示的是集中式的工作流机模型，右面是分布式工作流机模型，整个系统是一个由异构分布工作流机构成的工作流执行服务环境。对于工作流模型和工作流机集中，而工作流接口分布的工作流管理系统的结构，所有计算机上的活动执行由一个工作流机来控制。而对于由多个工作流机协作执行一个过程实例这种情况，被控制的过程实例的控制数据必须是这些不同的工作流机都可以访问的。控制数据可以集中存放在一个主机上作为一个共享资源使用，也可以将它分布到不同的工作流机环境中。在将控制数据分布到不同的环境中时，必须定义一套机制来保证这些控制数据之间的一致性。有关分布数据的一致性问题是多数据库管理系统中最常见的维护问题，在具体实施时，用户可以根据实际需要采用不同的解决方案。

2.2.3 工作流管理中的人机交互

在工作流管理系统的运作过程中，人和应用是完成整个业务过程的主体。工作流定义工具、工作流执行服务和任务表管理器都是为完成业务过程和支持人员工作提供的运行环

境和工具。具体说来，在整个工作流执行中，不同的操作人员需要完成的工作大约可分为以下几种：

1) 模型定义：创建、修改和发布企业的业务过程模型，这一般是企业的业务管理部门（如企管处）的人员按照企业业务流程完成；
2) 人机交互：按照工作流任务管理器提供的任务项，完成具体的业务处理工作（如填写表格、启动一个应用来计算生产计划、查询库存情况等），这个工作由企业的各个业务部门的人员完成；
3) 系统运行状态监控：检查、监视系统的执行情况，对于系统中出现的意外情况进行紧急处理，如终止、恢复某个过程实例的执行，改变某个活动的状态以便整个系统能够继续执行等。这个工作由具有较高职务的系统管理人员来完成。

2.2.4 工作流管理与群件

群件（GroupWare）是一种计算机软件系统，它的含义和覆盖范围相当广泛，所有能够支持工作组（规模可大可小、功能可强可弱、自动化程度可高可低）内成员协同工作的软件都可以叫做群件，同样所有支持工作组协同工作的技术都可以称为群件技术。群件根据其功能可以划分成以下几类：

- 消息系统：如电子邮件、电子公告牌系统、或者基于消息的计算机会议系统；
- 多用户编辑系统：支持实时的或者异步的共同写作；
- 群决策支持系统或者电子会议室：支持同步的群组交互；
- 实时电视会议系统、计算机远程会议和桌面会议系统；
- 智能代理系统：使能群组在复杂的信息结构下的导航；
- 协调系统：它们又分成：
  - 面向表单的系统：基于过程模型传递表单；
  - 面向过程的系统：通过编程技术实现过程集成；
  - 基于语言行为理论的面向会话的系统；
  - 面向通信结构的系统：基于角色关系来描述组织活动。

典型的群件技术有 CSCW(计算机支持的协同工作)技术。典型的群件产品如 IBM 的 Lotus Notes, Microsoft Exchange 等。工作流管理在一部分技术上继承于群件，而群件也融合了一些工作流的概念，提供了工作流的能力。图 2.6 从支持通信（communication)、支持

协作（cooperation）与支持协调（coordination）的 3C 角度出发对群件系统进行了分类。

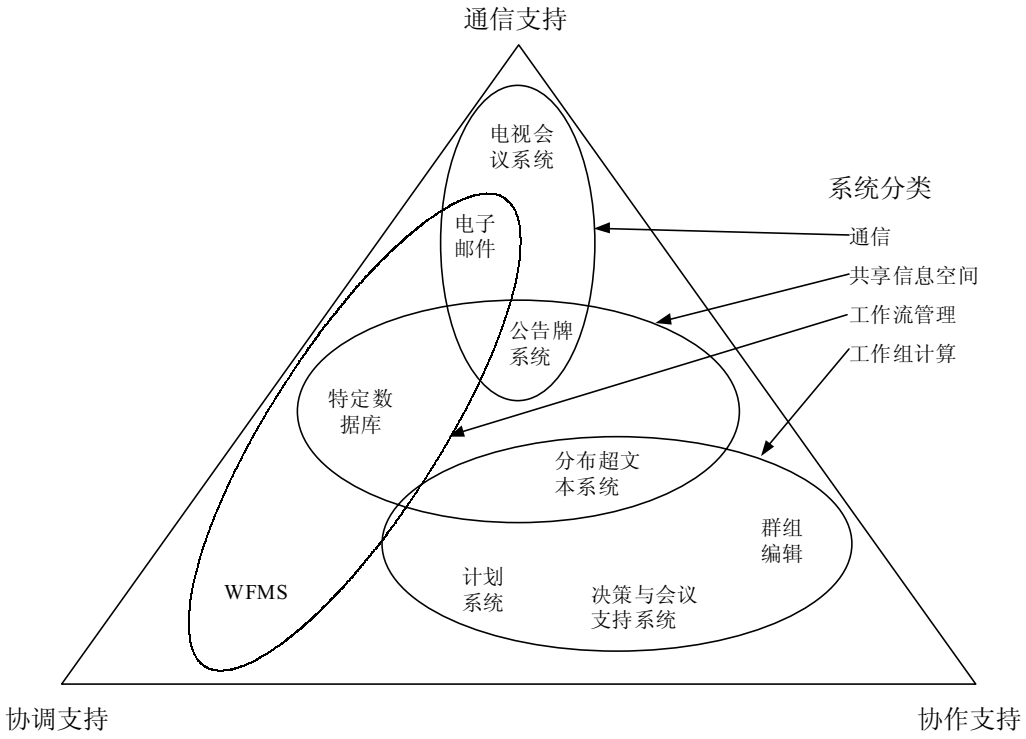


图 2.6 群件系统分类

群件和工作流之间没有明确的界限，许多人认为工作流也属于广义上的群件软件。我们认为虽然可以将工作流理解为属于群件软件，但是它与普通意义上的典型群件 Lotus Notes、Microsoft Exchange 等还是有一定的区别。它们的共同点都是支持群体合作，而不同点至少有以下一些：

1) 二者的侧重点不同。工作流管理系统要解决的是清晰地定义经营过程，并通过实例化来运行这个过程。而群件的重点是解决工作组成员之间的协作、共享和交互；

2) 群件提供了一定的工作流管理能力，但是不能说群件可以解决工作流管理。群件通过文档路由或电子邮件确实可以使工作从一个个体“流”向另外一个个体，但是群件缺乏严格意义上的工作流管理的大部分功能，如图形化的过程建模工具，对过程的监控，条件路由和异常处理等等，因此，不能将提供工作流能力的软件和工作流管理系统等同起来；

3) 目前成熟的群件产品大多提供了二次开发工具（如 Lotus Notes 的公式和脚本，Microsoft Exchange 的 Visual Basic），利用它们可以编制具有工作流能力的程序，但是，如果要产生一个简单的工作流应用就需要在编制程序方面进行大量的投资，甚至这个投资超过使用一个更具灵活性的工作流管理系统上的投资，就很不值得了。

2.3 工作流管理系统分类

在进行工作流管理系统分类之前，先对企业经营过程中处理的不同业务工作按照重复性和相应任务的结构化程度进行简单的分类。首先按照过程的重复性可以将经营过程分为周期重复的、无规律重复的和唯一的。

- **周期重复的：**经营过程的执行过程完全按照固定的流程执行，如银行申请贷款的业务，学生入学注册过程等。
- **重复的，但不是有规律的：**这种经营过程经常需要执行，并且其完成任务的目的是几乎相同的，但是由于任务的内容不同将导致处理过程略有区别。如处理用户对产品质量的投诉和处理过程等。
- **唯一的任务，仅在特定形式下发生一次：**这种任务一般仅在特定的情况下发生一次，如某个新桥梁、建筑的设计，建立一个新的生产车间等。

按照工作任务的结构化程度可以将经营过程分类为完全结构化的、半结构化的和非结构化的：

- **完全结构化的：**这种经营过程的执行逻辑完全可以事先确定，即可以对其制定严格的工作计划，一旦这种经营过程投入运行，它将严格按照事先确定的逻辑顺序执行。如银行申请贷款业务、学生入学注册、产品编码审批发放过程、产品入库过程等。
- **半结构化的：**这种过程的逻辑一部分可以预先确定，而其中有一部分逻辑无法事先确定，它们需要根据实际执行过程中的具体情况确定，另外一种半结构化的过程是过程的基本结构（整体框架）可以事先确定，但是其中某些具体的任务逻辑需要根据实际执行情况动态确定。
- 非结构化的：这种经营过程的活动顺序无法事先确定，如新产品类型设计，产品生产销售趋势分析等过程。

根据结构化程度和重复性可以将企业业务流程分成表 2.1 所示的 9 类过程。其中工作流管理系统最适合于应用于有规律重复的结构化过程。采用先进的柔性工作流管理系统也能够处理半结构化的和唯一的经营过程。而项目管理系统最适合于应用在唯一的结构化过程场景中。图 2.7 给出了不同类型的工作流管理系统的应用范围。

表 2.1 企业经营过程的特性分类

结构化程度	结构化	半结构化	非结构化

重复性			
有规律重复	每周财务报告、年度损益表	月度销售报告、给股东的年度报告	分析预计库存和实际存货的差距
无规律重复	记录收到的货物、更新库存	对于例外货物的订单处理	R&D 报告，设计新的啤酒馆装饰
唯一	法人形式改变	建立新的生产厂	新产品类型设计

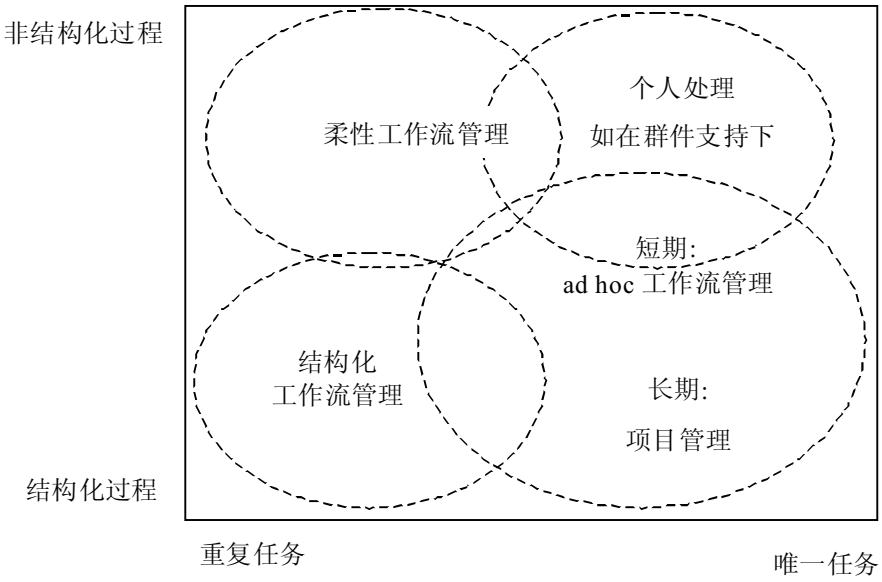


图 2.7 不同工作流管理系统的应用范围

综合考察目前的工作流产品市场，还可以根据所实现的业务过程、底层实现技术及任务项传递机制这三种分类方法对工作流管理系统及其产品进行分类。根据所实现的业务过程，工作流管理系统可分为四类：

1) 管理型工作流（administrative workflow）：在这类工作流中活动可以预定义并且有一套简单的任务协调规则，例如，大学里的课程选修，完成论文后的学位申请等。
2) 设定型工作流（ad hoc workflow）：与管理型工作流相似，但一般用来处理异常或发生机会比较小的情况，有时甚至是只出现一次的情况，这与参与的用户有关。
3) 协作型工作流（collaborative workflow）：参与者和协作的次数较多。在一个步骤上可能反复发生几次直到得到某种结果，甚至可能返回到前一阶段。
4) 生产型工作流（production workflow）：实现重要的业务过程的工作流，特别是与业务组织的功能直接相关的工作流。与管理型工作流相比，生产型工作流一般应用在大规模、复杂的和异构的环境下，整个过程会涉及许多人和不同的组织。

根据底层实现技术， 可将工作流产品分为三类：

1) 以通讯为中心：以电子邮件为底层的通讯机制。这种类型的工作流管理系统适合于协

作型工作流和不确定型工作流，而不适于生产型工作流。

2) 以文档为中心：基于文档路由，它同外界应用的交互能力有限。许多基于表的管理型工作流可以用以文档为中心的工作流实现。

3) 以过程为中心：这种工作流系统对应生产型工作流。它们一般建立在数据库之上，有自己专用的通信机制并且提供了同外部进行交互的接口。

根据 Delphi Group 的报告，1997 年度，在所有的工作流产品中，以过程为中心的产品占据了 54%的市场，以文档为中心的占 32%，以通讯为中心的产品占 14%。过程类产品发展历史最长，但目前发展速度最慢，比较新的文档类产品则相对发展较快，而最近才出现的通讯类产品发展速度最快。五年以前，工作流产品的市场几乎被过程类产品所独占，而在 1997 年，新出现的两类工作流产品却占据了 46%的市场。

Delphi Group1997 年的调查结果显示，在各个行业中，商业（包括银行业和保险业）依然是工作流产品的最大用户。政府行政部门居其次。制药业和卫生业超过了制造业，成为影响工作流产品的第三大类行业。通讯业也在三年中从 2%显著增加到 1997 年的 5.7%，增长率超过了 165%。

根据不同工作流系统所采用的任务项传递机制的不同，市场上的工作流产品又可以划分为四类[10]：

1) 基于文件的工作流系统——以共享文件的方式来完成任务项传递。这种类型产品开发得最早、发展最成熟、其产品品种较多。代表产品有 FileNet 的 Visual WorkFlo、IBM 的 FlowMark、InConcert 的 InConcert。

2) 基于消息的工作流系统——通过用户的电子邮件系统来传递文档信息。这种类型的产品一般都提供与一种或多种电子邮件系统的集成接口。代表产品有 Novell 与 FileNet 合作开发的 Ensemble、JetForm 公司的 InTempo、Keyfile 公司的 Keyflow。

3) 基于 Web 的工作流系统——通过 WWW 来实现任务的协作。这一类产品起步较晚（在 95 年以后），但是发展迅速，其市场前景十分看好。许多供应商纷纷改进原有产品或开发新产品以增加对 Web 的支持。代表产品有 Action Technologies 公司的 ActionWorks Metro、Ultimus 公司的 Ultimus。

4) 群件与套件系统——虽然这一类产品与上面介绍的三种产品在任务传递方式上有很大程度的重叠，但是在这里却有必要把它们单独划分成一类，因为这一类产品都需要依赖于自己系统的应用基础结构，包括消息传递、目录服务、安全管理、数据库与文档管理服务等，它们本身就构成了一个完整的应用开发环境。代表产品有 IBM/Lotus 公

司的 Lotus Notes、Microsoft 公司的 Office 与 Exchange、Novell 公司的 GroupWise。

2.4 工作流管理系统的实施

2.4.1 工作流管理系统的实施

工作流管理系统不同于 ERP 和普通的企业管理信息系统，ERP 与普通的企业管理信息系统是事务处理系统，其主要目的是满足企业业务操作功能，提高企业事务处理的效率和水平。从企业整体的业务流程和企业经营目标上看，事务处理系统一般局限于解决某个或者某些领域的问题；事务处理系统的另外一个局限性是它一般局限于解决企业内部的具体操作问题，面向企业内部功能，而不是面向市场和面向客户的系统。工作流管理系统的着眼点是面向市场、面向客户，其目标是在整个企业的业务层提高企业的业务处理水平、强化企业的市场意识、提高对市场的应变能力。

由于工作流管理系统与普通事务处理系统存在显著的差别，工作流管理系统在企业的实施方法上也不同于普通的事务处理系统。要实施工作流管理系统首先要在战略层次上对企业的业务目标进行分析，确定企业的战略目标和组织要求。图 2.8 给出了工作流管理系统实施的层次结构。

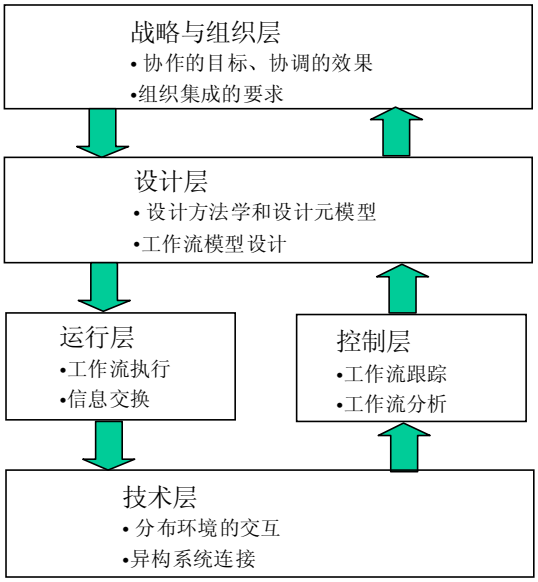


图 2.8 工作流管理系统实施的层次结构图

在完成了企业战略目标分析和工作流实施战略后，工作流管理系统才能够进入真正的

实施阶段。工作流管理系统在实际系统中的应用一般分为 3 个阶段，即模型建立阶段、模型实例化阶段和模型执行阶段。图 2.9 给出了工作流管理系统的应用的三个阶段。模型建立阶段通过利用工作流建模工具完成企业经营过程模型的建立，将企业的实际经营过程转化为计算机可处理的工作流模型。模型的实例化阶段完成为每个过程设定运行所需的参数，并分配每个活动执行所需要的资源（包括资源、人员、应用）。模型执行阶段完成经营过程的执行，在这个过程中重要的任务是完成人机交互和应用的执行，并对过程与活动的执行情况进行监控与跟踪。

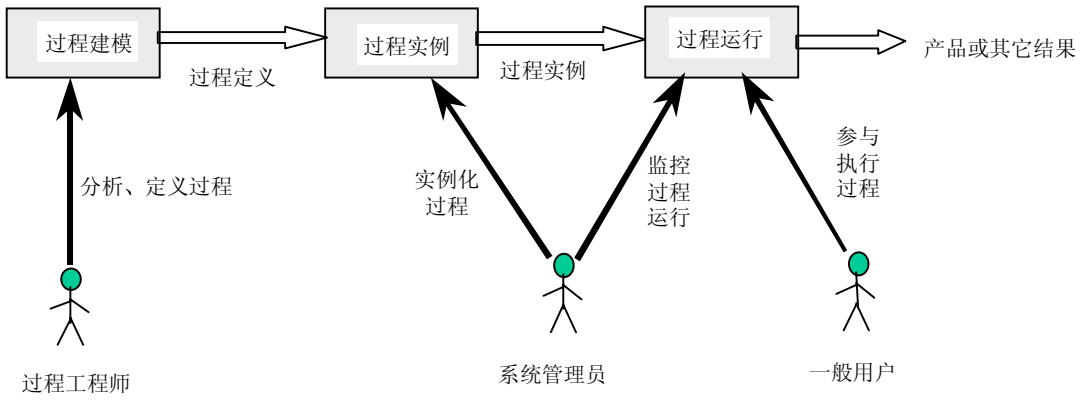


图 2.9 工作流管理系统实施的三个阶段

在实际应用中，工作流管理系统的实施与企业的经营过程重组与经营过程改变是紧密相关的。实施工作流管理系统的目的就是要提高企业的柔性，并且能够根据市场的变化不断改进其业务过程，因此其相应的工作流模型也需要不断的进行改进。

前面已经指出，工作流管理系统的实施不同于普通的 ERP 或者 MRPII 系统，工作流管理系统的实施是一个不断循环，不断改进的过程，这个特性使得工作流管理系统的实施和应用在柔性和可扩展性上要远远优于普通的管理信息系统。图 2.10 给出了工作流管理系统实施的循环图。图中数字 1 到 6 基本上表示了工作流实施过程的一个循环。其中工作流结构与路由数据是一切工作的信息基础。下面针对图中给出的数字顺序简单介绍一下实施的流程。

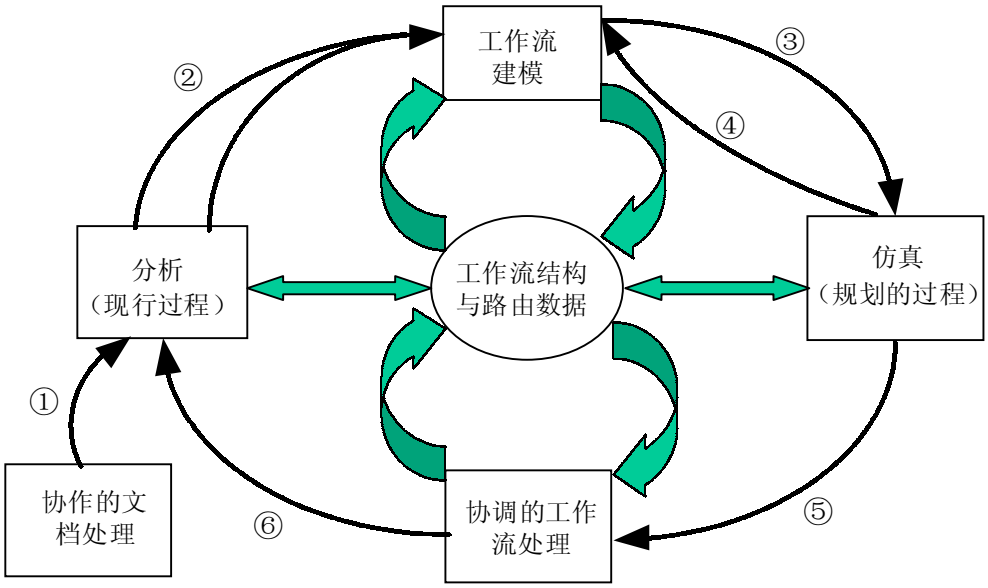


图 2.10 工作流管理系统实施的循环图

① 通过对企业现有的业务流程、业务规则、管理规章等进行分析，总结得到企业的现有业务流程模型（这时的模型还不是工作流模型）；

② 采用 BPR 工具对现有的业务流程模型的分析，利用工作流建模工具或者其它 BPR 工具对现有的流程经过抽象与整理，得到现有系统的工作流模型；

③④ 对该工作流模型进行仿真，找到存在的问题，并结合企业的战略目标或者经营思想、市场情况、客户需求，对模型进行改进，得到优化的业务过程的工作流模型；

⑤ 将该模型投入实际运行，得到运行的结果；

⑥ 对运行的结果进行分析，发现存在的问题，进一步改进工作流模型。

需要指出的是，在上述循环中，创建或者改进过程模型可以在任何时候进行，如企业可以不在现有流程的基础上，而是根据一个新的市场需求直接建立一个工作流模型，然后进行仿真。

2.4.2 采用工作流管理系统的好处

工作流管理的目的是根据预定的目标，找到合适的方法和手段来优化工作流程。其在企业的应用会给企业带来巨大的效益。首先，采用工作流管理将使企业改变其传统的按照功能来配置其人员的组织结构，变成按照企业要实现的主要业务流程来配置组织结构，这样可以大大缩短其主要业务过程的处理时间，提高其对市场的响应能力。组织结构的改变将大大减少在企业内部不必要的物料、信息的传递时间。当然，整个企业组织结构的调

整首先需要调整传统的以部门组织生产、人员从属于某个部门的做法，变成以项目来组织生产和人员的工作方法。一个人可能同时从属于多个项目。

在企业进行工作流管理系统应用主要可以取得有效好处：

- 提高企业管理的规范化程度；
- 更好地与上下游企业形成快速响应市场的供应链网络；
- 降低业务过程的整个处理时间，如在办公自动化环境中，通过更好地规划工作流程、并行执行相互独立的活动、减少文档传递过程中不必要的中间状态等方法来显著降低文档的传递和临时储存的时间。
- 降低管理成本，如避免不必要的和重复的工作，提高工作人员的工作效率；
- 改进工作质量，如自动提供为完成某个任务所需要的相关信息。在客户服务中，能够快速方便的访问所有相关数据和工作流程，从而可以大大提高客户服务质量；
- 在工作人员之间更好地均衡负荷，如在工作人员缺勤的情况下，自动地柔性分配替代人员；
- 通过在工作流模型加入对可预计的故障的处理策略来提高系统的柔性；
- 在工作流的基础上改进控制策略，降低相应的控制成本；如通过监控工作流执行状态，利用分析和控制工具来进行优化控制；
- 通过对已经完成的工作流实例的分析，找出存在的不足，进而不断改进工作流程；
- 使工作内容更加丰富，并且提高工作人员的业务能力，减少工作人员进行单调乏味并且十分耗时的文档查找工作。

除了上面给出的从企业的经营目标出发可以得到的好处外，采用工作流管理系统对于企业的信息的现代化同样具有显著的作用。采用工作流管理系统可以在最大程度上集成企业的现有信息资源，实现资源的充分利用。由于工作流管理系统具有较好的柔性和开放性，因此可以保证企业的信息系统能够顺利的进行扩展以满足不断变化的市场环境。另外，由于工作流管理系统是在工作流模型的基础上进行企业的业务过程运行，这就意味着企业的信息系统运行已经从过去没有一个具体的可量化指标的管理信息系统，发展到了一个建立在工作流模型上（并且是可以利用 BPR 或者其它仿真工具进行优化后的模型），按照首先定义好的规则进行执行，并且对于执行的结果随时进行监控和评价的规范化阶段。这种由过程建模－－>模型分析－－>过程优化－－>模型执行－－>执行结果统计分析－－>改进业务过程－－>优化企业运作的实施方法为成功地实施企业信息系统奠定了坚实的基础。

2.5 工作流产品与研究情况

2.5.1 现有工作流产品存在的不足

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，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水平。在工作流管理系统的开发的最初阶段缺乏统一的普遍遵循的标准以及限于当时软件支撑技术的水平，导致不同公司的工作流管理系统在功能上、所采用支撑技术上、开发技术上和接口上都呈现出非常大的不同。这也是目前工作流管理系统存在问题的主要原因之一。目前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——也就是说，工作流管理系统必须要建立在适当的底层通讯基础之上，以便实现执行工作流所需要的分布计算环境。比如 CORBA、DCOM、Java 都是可以选择的。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。这样，企业若想布署工作流系统，还必须要额外付出经费与精力去构筑底层环境，这种情况是企业所不希望的，实际上它也限制了工作流管理系统在企业的广泛应用。

（2）缺乏标准——不同的厂商所提供的工作流产品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。在缺乏标准的情况下，用户一旦选定一种产品之后，就很难再过渡到其它同类产品之上了；而且不同的系统之间缺乏互操作的接口，使得用户有很多后顾之忧。尽管工作流管理联盟（WfMC）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。目前，大多数工作流产品的生产厂家已经意识到了这个问题。他们加入了工作流管理联盟，并提出了一套工作流管理系统的标准。主要工作包括：开发了一套 API，保证以一致的方式访问工作流管理系统的服务和功能；为工作流管理系统之间和管理系统与应用程序之间规定了交换格式和协议；统一了工作流模型，使得不同工作流管理系统的工作流定义可以相互使用。

（3）实现的复杂性——将工作流管理系统在企业进行应用不仅仅需要完成过程的定义，还

需要做许多其它工作，总的说来工作流管理系统在企业的应用是一个复杂的实施过程。这个复杂的实施过程中需要完成的任务包括：对外部的应用系统进行包装以便工作流机能够在必要的时候激活它、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、还包括制定相应的管理规程和用户操作规范等等。目前的工作流产品为这些任务的完成提供的帮助能力有限，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，同时还会对企业原有的管理制度造成一定的影响，因此，企业需要完善相应的管理体系，以便工作流系统能够在最大程度地发挥其作用。

（4）系统的安全性－－目前的工作流管理系统对系统运行中出现的并发访问和异常错误缺乏正确和可靠的支持。工作流实例在运行过程中，有可能出现多个活动同时访问共享资源的情况，各个活动在数据操作上会相互重叠。为了保证系统能够正常运行，必须要进行并发控制（Concurrency Control），防止出现“脏数据”等现象。另外，当工作流在执行过程中出现非正常中断时，如何恢复数据并保持数据的一致性也是需要解决的问题。目前大多数工作流产品在这两方面的性能还有待进一步增强。

（5）性能问题－－目前大多数工作流产品无法满足企业对每天处理上万个，甚至每小时处理几千个业务的需求。

（6）工作流技术尚不成熟——尽管工作流技术从最初萌芽发展到现在已经有了很大的进步，但工作流技术并不成熟。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原来使用其它技术（如数据库）实现的关键业务应用进行重组。

不少从事工作流技术研究的人员从应用的角度指出了目前工作流系统存在的性能方面的不足，在文献[11]中，Mohan 等人指出现有工作流管理系统存在的主要不足在于：几乎彼此都不兼容；能力有限，对大规模业务的负荷无法胜任（主要因为单一数据库、集中式结构、通讯能力有限、设计时缺乏远见等原因所致）；鲁棒性与实用性差（系统稳定性差，没有强有力的安全保证）。

根据我们对工作流技术的研究认识，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）工作流模型描述：缺乏一种能够支持过程定义、过程演进以及过程分析的形式化数学

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顺序关系。目前的工作流模型大部分都是从直觉出发，以图形语言或者文本语言来定义工作流过程，这种定义的方法本质上还停留在用户层上，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，因而也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[12]，但从其模型所具有的描述能力上距离对工作流过程本质的描述上仍有差距，具体表现在模型元素不丰富、动态性能不理想、仍离不开图形语言的局限。由于模型缺乏良好的理论基础，使得工作流系统在应用的许多关键特性上无法得到保证，包括过程模型的柔性定义、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普及与推广。

（2）工作流执行：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，制约工作流应用中的一个很严重的瓶颈问题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。另外，工作流机的分布式执行也是一种必然的发展趋势，但同样也带来一些需要解决的复杂的问题，比如不同工作流机之间的协调问题、分布事务处理问题、系统中各类数据的维护问题、安全问题等。只有工作流机真正实现可靠、稳定、高性能的分布，才能提高工作流系统在企业应用中的实用性。

（3）工作流仿真：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难以确定、仿真的过程较为复杂。

2.5.2 工作流管理技术的主要研究问题

在工作流技术应用日益得到重视的今天，对工作流技术的研究也正在向更深层次进行。工作流技术研究的主要目的有两个：一是为工作流技术的发展解决理论上存在的问题，探讨工作流模型和语义的形式化表示方法等；二是从工作流实现技术的角度探讨利用先进的技术提高工作流管理系统的性能和可靠性。

工作流技术研究的一些理论研究问题为：

1) 过程建模理论与建模方法：研究如何清晰、准确地表示实际应用中的过程，特别是研究如何以形式化的方法表示过程模型。如采用 Petri 网方法来表示工作流模型、定义工作流过程描述语言（尤其是形式化语言）、采用 EPCM 等，这方面的研究工作和取得的一些成果参见[7]。研究和发展能够支持事务的工作流模型可以从根本上提高工作流系统的正确性和可靠性，因此关于事务工作流模型的研究也得到了充分的重视。
2) 模型验证和模型仿真方法：研究从理论上如何验证所建立的过程模型不存在死锁问题，研究如何评价所建立的模型的性能和优化模型的方法，这个问题对于企业经营过程重组问题的研究和实施企业经营过程重组具有重要的意义。
3) 分布环境下的资源协调和任务优化调度策略：研究在大范围的分布环境下，在资源有限（如制造系统中只有有限的加工设备和运输设备）和任务完成时间约束（如产品交货期要求）情况下，如何优化系统中任务执行的顺序和资源的分配方法，这个问题的研究对于工作流技术在大规模生产和调度中有非常重要的意义，如在供应链管理中的最佳供应和分销策略，复杂的交通管制中的应用等。它对于解决过程模型执行过程中产生的冲突（如并行工程中的产品设计过程中的约束冲突）也有重要的意义。
4) 过程模型与其它模型的集成方法：在描述一个企业和一个应用领域的问题时，仅有过程模型是不够的，还需要有功能模型、信息模型、资源模型、组织模型、甚至经济模型和决策模型等的配合，这些不同的模型描述了一个应用领域的不同侧面，它们的集成可以完成对一个企业或一个应用领域的全面描述。但是如何集成这些模型还没有良好的解决方法。有些工作流建模工具中包括了对资源和组织的描述能力，但是目前还没有一个方法来实现过程模型和功能、信息模型的集成。这方面的研究工作对于促进集成化的企业建模和信息系统实施有重要的意义。

在工作流管理系统的实施上也存在非常多的问题需要研究，这些问题的解决对于提高工作流管理系统的性能具有重要的意义。

1) 分布式工作流机的实施问题：在通常情况下，企业的应用一般都是运行在分布式环境上，因此相应的工作流机也应该采用运行在分布环境下的分布式结构。目前已经开发出了许多分布式工作流管理系统，如 IBM 的 Exotica、佐治亚大学计算机系开发的 Meteor 等。但是仍然有许多问题需要进行深入的研究，如工作流模型实例到每个执行工作流机的分解问题、分布式环境下的可靠性问题，分布式系统的效率和鲁棒性问题等。在 CORBA 技术和 WEB 技术得到广泛重视的今天，研究开发在 CORBA 环境下，WEB 页面方式的工作流管理系统具有非常重要的应用价值，佐治亚大学计算机系开发的 Meteor 在这个方面取得良好的成果。分布环境下的工作流机的合作问题也是受到广泛关注的重要问题，它对于提高工作流管理系统的性能具有重要的意义。
2) 异常处理和错误恢复问题：对于制造企业这样的一个复杂的应用系统，出现异常和错误是非常正常的情况，对于工作流管理系统不仅要求它在正常情况能够发挥作用，更重要的是要求它能够灵活的处理各种异常情况，并且在某个节点发生错误时能够保证整个系统不会发生崩溃。目前大多数工作流管理系统都提供了异常情况处理策略、系统容错能力、故障恢复策略、数据保护及数据恢复方法。
3) 融入事务管理概念：在工作流管理中融入事务管理的概念和模型方法可以显著的提高工作流管理系统处理大规模业务应用的能力，目前这个工作已经得到许多研究人员的重视。
4) 应用集成问题：工作流管理系统本身是一个完成过程建模和过程管理的软件系统，但是为了在企业的实际业务中得到有效的应用，它必须和企业已有的或购买的其它实现业务应用的软件实现集成，通过集成来提高整个企业的应用水平和应用效率。

第 3 章 工作流管理系统参考模型

3.1 工作流管理系统体系结构

工作流管理是正在迅速发展的技术，它在不同的行业已经得到了应用。工作流管理技术与工作流管理系统得到广泛重视的一个重要原因是它能够在信息技术的支持下实现基于人工和计算机活动组成的业务过程的自动化，它可以实现不同自动化程度（人工操作、半自动化、自动化过程）的规范化业务管理功能，具有良好的适应性。因此，虽然工作流管理最早是在办公自动化领域（保险业、银行、法院、行政管理）开始进行应用的，它在工业领域的应用同样取得了显著的成果，尤其是在制造领域得到了广泛的应用。

随着对工作流产品需求的不断扩大，许多公司纷纷推出了不同的工作流产品。这些工作流产品都有自己的特点，也有自己的协议和接口标准，它们在不同的应用领域进行了应用。本书的第四、五章将对现有的一些主要的工作流研究和产品情况进行介绍，并对它们进行一些比较。但是由于工作流管理技术与产品缺乏统一的标准，这些不同的工作流产品从术语的定义和使用、系统结构的设计到与应用之间的接口规范上都存在较大的差异，导致这些产品之间、产品与其它应用之间的集成十分困难。按照对系统开放性的要求，这些工作流系统和产品的规范化程度和开放性不够，导致它们之间不能够实现互操作。工作流管理系统互操作是指两个或多个工作流机之间通讯和协作工作的能力，具有通讯和协作的能力就称为可以互操作，否则就称为不能互操作。不同工作流管理系统之间不能互操作这种情况给开发商和用户都带来了很大的不方便，也在一定程度上阻碍了工作流管理系统的推广和发展。

为了能够更好的支持企业经营过程建模、分析和实施，适应世界市场的多元化趋势，需要建立工作流管理系统的相关标准，从系统结构、术语使用、接口实施方面提供标准化与规范化的定义，并以此为基础实现不同工作流产品之间的互操作，方便与其它应用系统的集成。在建立工作流的相关规范和标准方面，国际上成立了一个称为“工作流管理联盟”（简称 WfMC）的国际组织。它提出了有关工作流管理系统的一些规范，定义了工作流管理系统的结构及其与应用、管理工具和其它工作流管理系统之间的应用编程接口，其主要目的是为了实现工作流技术的标准化和开放性，从而支持异构工作流管理系统与产品之间的互操作，并且使得其它的应用可以使用该结构和定义好的通用 API（应用编程接口）访问不同的工作流管理系统提供的服务，实现与其它应用的快速有效集成。

图 3.1 为 WfMC 提出的工作流参考模型的体系结构图。这个参考模型的体系结构给出了抽象的工作流管理系统的功能组成部件和接口，它能够满足工作流管理系统和产品所应该具有的主要功能特征，可为实现工作流产品之间的互操作提供公共的基础。必须指出，组成工作流管理系统的每个功能部件可以在不同的软硬件平台上采用不同的方法实现，同样接口也可以在不同的软硬件平台上采用不同的设计技术和编程语言进行编程。一般说来，工作流产品的提供商也不会将这些部件之间的所有接口完全对外开放，但是为了实现不同工作流产品之间的集成，它们会按照互操作和协作的不同要求在一定层次上开放其接口。

从图 3.1 可以看出，工作流管理系统主要由三类构件组成，这三类构件分别是：

1) 软件构件：完成工作流管理系统不同组成部分功能的实现；

2) 系统控制数据：工作流管理系统中的一个或多个软件构件使用的数据；

3) 应用与应用数据：对于工作流管理系统来说，它们不是工作流管理系统的组成部分，而是属于外部系统和数据，它们被工作流系统调用来完成整个和部分工作流管理的功能。

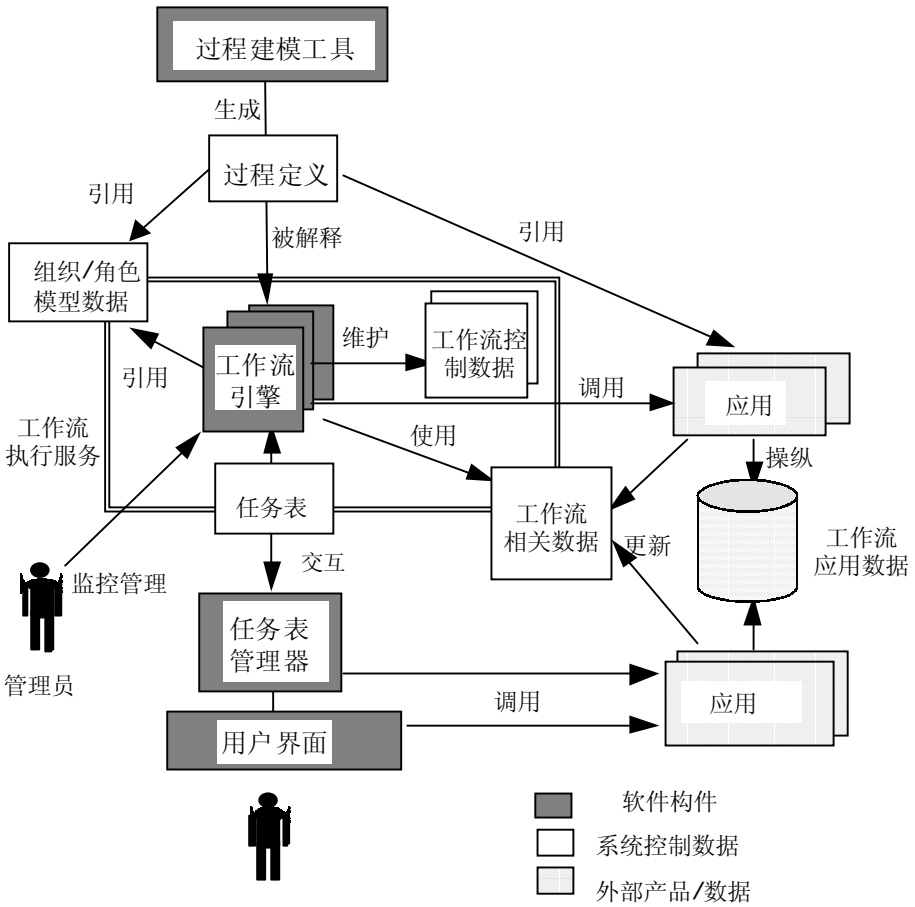


图 3.1 工作流管理系统的体系结构图

3.2 工作流参考模型

图 3.2 给出了 WfMC 提出的工作流参考模型。本节首先介绍工作流参考模型中涉及到的几种数据，在以后的各节对系统中的各个部分和参考模型中的五类接口进行描述。

工作流控制数据（Workflow Control Data）：工作流执行服务/工作流机通过内部的工作流控制数据来辨别每个过程或活动实例的状态。这些数据由工作流执行服务/工作流机进行控制。用户、应用程序或其它的工作流机/工作流执行服务不能对其直接进行读写操作，它们可以通过向工作流执行服务/工作流机发消息请求来获得工作流控制数据的内容。

工作流相关数据（Workflow Relevant Data）：工作流管理系统通过工作流相关数据来确定过程实例状态转换的条件，并选择下一个将执行的活动。这些数据可以被工作流应用程序访问并修改。因此，工作流管理软件需要在活动实例之间传递工作流相关数据。

工作流应用数据（Workflow Application Data）：这种数据指那些由应用程序操作的数据。它们是针对应用程序的，是企业完成具体的业务功能所需要的数据，如产品结构数据、订单数据、生产作业计划数据等。工作流管理系统无法也不需要对它们进行访问。

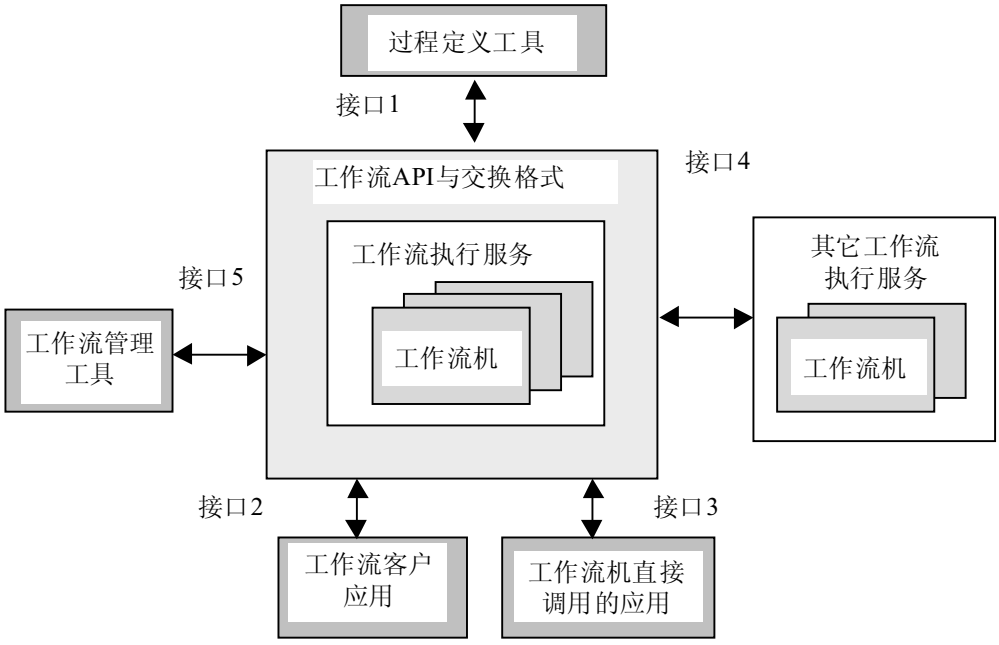


图 3.2 工作流参考模型

3.3 工作流模型和建模工具

工作流模型包含了描述一个能够由工作流执行服务软件系统执行的过程所需要的所有

信息。这些信息包括过程的开始和完成条件、构成过程的活动以及进行活动间导航的规则、用户所需要完成的任务、可能被调用的应用、工作流机的引用关系、以及所有与工作流相关数据的定义。过程定义可能引用组织/角色模型中关于组织结构、组织中的角色等信息。这样在进行过程中活动或信息对象的定义时，不仅可以指定某个特定的人是这个活动或信息对象的参与者，而且可以将活动或信息对象与组织实体或角色功能进行关连。过程定义指定完成某项活动的组织实体或角色，而不是定义具体的人员。在工作流执行环境中，工作流执行服务负责将组织实体或角色功能与特定的参与者进行连接。这样做的好处是不仅可以增加过程定义的方便性，而且还可以提高工作流执行服务的灵活性。

我们以本书第 2 章例 1 描述的某个电脑公司的计算机销售过程为例来说明组织实体和角色功能如何与活动进行关连的。在例 1 中，活动“用户资格审查”由销售部门的“用户资格审查员”这个角色完成，而在销售部门中有职责或资格作为“用户资格审查员”的可能有多个人员，比如说销售经理、销售财务、用户管理人员等。活动“决策”通常由公司“业务主管”这个角色来完成，而“业务主管”这个角色可以是公司副经理以上的人员来担任。另外，我们还可以定义活动“用户资格审查”是由销售部门的资格审查科完成，这样我们就完成了活动与这个组织单元的关联，在活动“用户资格审查”需要执行时，由资格审查科完成其操作。

过程建模工具以计算机能够处理的形式进行过程的定义。它可以基于形式化的过程定义语言、对象—关系模型来进行过程模型定义。对于比较简单的系统，也可以采用一组路径选择命令的方式来描述信息或文档在不同参与者之间的传递过程。过程建模工具可以作为特定工作流产品的一部分提供给用户，也可以作为一个单独的产品提供给用户，它还可以作为企业经营过程分析系统的一部分提供给用户。在作为企业经营过程建模分析软件产品提供给用户时，这个过程建模工具应该提供一个由分析模型到投入实际实施的运行模型的转换接口，从而使得由这个工具生成的模型能够被企业实际应用的工作流管理系统执行。

工作流建模工具应输出一个能被工作流机解释并执行的过程定义。不同的工作流产品其建模工具输出模型的存储格式是不同的，图 3.2 参考模型中接口 1 的定义不仅是为了实现工作流的定义阶段和运行阶段的分离，使用户可以分别选择建模工具和工作流执行的软件产品，而且是为了使不同的工作流产品能够实现协作运行，从而为过程定义的执行提供良好的运行服务环境。

在工作流建模上，工作流管理联盟开展了两个方面的工作：

1) 定义了一个元模型：所谓元模型一般是指描述模型的模型。这里的工作流模型的元模

型是用来描述工作流模型内在联系的模型。它用于描述工作流模型内部包含的各个对象、对象之间的关系及对象的属性，这个元模型有利于建立可以在多个工作流产品之间交换信息的模型。

2) 定义了一套可以在工作流管理系统之间、管理系统与建模工具之间交互过程模型定义的API（应用编程接口）(接口1)。

图3.3为工作流管理联盟定义的过程元模型。在该模型中包含了以下几个基本实体：

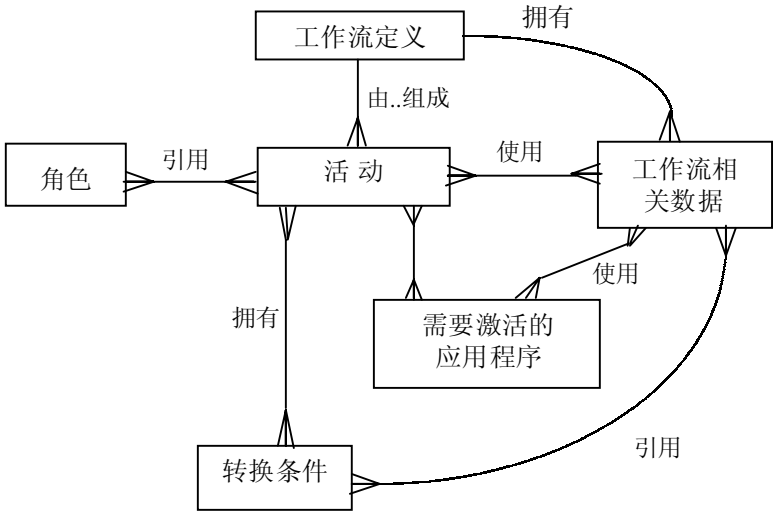


图3.3 过程定义元模型

1) 工作流定义（过程模型）：它一般包含诸如工作流模型名称、版本号、过程启动和终止的条件、系统安全、监控和控制信息等一系列基本属性。这个过程模型反映了企业中一个经营过程的目的，即这个过程要实现的目的和最终达到的目标是什么。

2) 活动：主要属性有活动名称、活动类型（原子级活动、子流程等)、活动的前、后条件、调度约束参数（如最长处理时间、排队等待时间等）等。当工作流运行在分布的环境下时，在活动的属性中还应该包括执行该活动的工作流机的位置。活动相应于企业经营过程中的任务，主要反映完成企业经营过程需要执行哪些功能操作。

3) 转换条件：主要负责为过程实例的推进提供导航依据，主要参数包括工作流过程条件（flow condition，过程实例向前推进的条件，可以认为是前/后条件的同义词)、执行条件（execution condition，执行某个活动的条件）和通知条件（notification condition，通知不同用户的条件)。转换条件对应于企业经营过程中的业务规则和操作的顺序。如在订单处理完成后，执行生产计划制定。

4) 工作流相关数据：工作流机根据工作流相关数据和转换条件进行推进，工作流相关数据的属性包括数据名称、数据类型和数据值等。它是工作流机执行任务推进的依据，

如在银行贷款申请表处理后，根据申请贷款的值（如是否大于10万元）决定下一个执行的活动是什么，比如大于10万元的申请交业务经理处理，小于10万元的申请交给业务员处理。

5) 角色：角色属性主要包括角色的名称、组织实体（organizational entity)、角色的能力等。角色或组织实体决定了参与某个活动的人员或组织单元。它主要描述企业经营过程中参与操作的人员和组织单位。

6) 需要激活的应用程序：主要属性包括应用程序的类型、名称、路径及运行参数等。应用主要描述了用于完成企业经营过程所采用的工具或手段。如采用ERP软件或决策支持软件完成某个具体的企业业务功能。

图3.3中给出的过程定义元模型的组成核心是活动。工作流定义与活动、工作流相关数据之间是一对多的关系，即一个工作流定义由多个活动与多个工作流相关数据组成。活动、角色、工作流相关数据、需要激活的应用程序、转换条件之间都是多对多的对应关系。如一个活动可以引用多个角色、使用多个工作流相关数据，同样一个角色可以被多个活动引用，一个工作流相关数据可以被多个活动使用。

在定义模型的交互格式方面，还需要有一套完整的命名机制来保证在工作流执行服务在运行期间，它所执行的所有过程与活动的名称可以准确的对应到实际的名称和地址。这个对应可以使用动态的地址解析机制（如使用目录服务）或其它机制来实现。

WfMC定义的工作流管理系统接口1描述了过程定义输入与输出接口。这个接口为在不同物理或电子介质之间传递过程定义信息提供了交互的形式和API（应用编程接口）调用函数。图3.4给出了过程定义的交换接口形式。

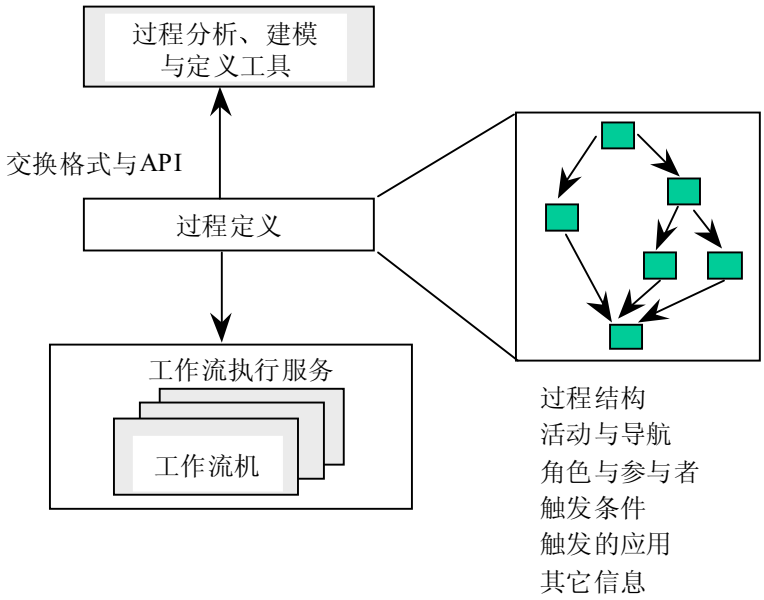


图3.4 过程定义交换接口

使用标准定义接口具有很多好处：首先，它可以实现建模环境和运行环境的分离，使一种建模工具创建的模型可以运行在多个不同工作流产品上；其次，它使得多个工作流产品可以实现协同工作，构成一个协作的工作流执行服务环境，这个执行环境可以运行由同一个过程模型所生成的过程实例。工作流管理联盟提供的API（应用编程接口）函数在建模方面主要覆盖了以下几个功能：

1) **通信建立：**建立各个参与运行的系统之间通讯连接，并且在完成协作后断开通信连接。
2) **工作流模型操作：**对过程模型的名称进行检索，并完成对过程模型对象的读/写操作等。
3) **工作流模型对象操作：**在建模工具中完成创建、检索和删除对象的操作，并可以完成创建、设置和删除对象属性的操作。

3.4 工作流执行服务与工作流机

工作流执行服务是工作流管理系统的核心，实际上它是企业经营过程的任务调度器，并且还在某种程度上是企业资源分配器。在采用工作流管理系统支持其经营过程运行的企业，工作流执行服务可以看成是企业的业务操作系统。企业的业务过程在它的管理、监控和调度下运行，因此工作流执行服务系统的性能和可靠性就直接决定了企业经营过程的运行效率和安全性。工作流执行服务由一个或多个工作流机（工作流机又称工作流引擎－Workflow Engine）组成（在分布环境下由多个工作流机组成），它提供了过程实例执行的运行环境，主要完成以下功能：

1) **实例化及执行过程模型：**解释企业经营过程的过程定义,根据过程执行需要的初始条件和执行参数生成过程实例,运行过程实例并管理其运行过程。这里需要指出的是一个过程模型实际是企业经营过程的一个模板，它可以被执行多次，也可以有多个有关这个过程模型的实例在同时运行。如订单处理过程，每当来了一个新的订单时，它都启动了一个新的工作流流程，只不过每个流程处理的订单不同而已。因此，运行多个订单处理过程模型的实例意味着有多个订单在被处理。
2) **为过程和活动的执行进行导航：**根据过程定义和工作流相关数据，为过程实例的运行进行导航，如根据过程的进入和退出的条件启动和终止一个过程实例；根据活动之间的关联和活动的执行条件，决定并行或串行执行后续活动；给用户提供需要操作的工作流任务项信息；或者根据所需激活的应用程序信息启动相应的应用程序等等。

3) **与外部资源交互完成各项活动：**工作流执行服务通过两种途径完成与外部资源和用户的交互：客户应用接口和直接调用应用接口方式。对于客户应用方式，工作流机通过任务项列表管理器对应用的执行进行管理。任务项列表管理器提供任务项列表供用户进行选择，并记录监督工作项的完成情况。由用户完成从任务项列表管理器提供的任务项列表中选择相应的任务项，并在需要的时候调用应用工具完成相应任务的执行，在任务执行完成后，用户需要修改相关任务项的状态，如置完成标志，供任务项列表管理器使用。这些通过任务项列表管理器分发并管理的需要用户操作的活动对应于工作流管理系统中用户手工完成的活动（非自动化活动），如在完成对一个产品招标标书的评审后，由业务员向供应商发出竞标成功的通知，并与供应商签订供货合同。

对于直接由工作流机启动的活动，由工作流机直接调用相应的应用来完成，这些自动执行的应用同样需要将合适的预先定义好的应用执行完成情况反馈给工作流机。工作流机自动调用的应用主要是针对基于服务器的无需用户参与的应用，即自动化活动。如在某个设计图纸完成电子会签后，自动进行版本发布并将图纸归档。

4) **维护工作流控制数据和工作流相关数据：**工作流在执行过程中要维护不同过程和活动实例的内部状态信息，以及用于协调和恢复的各种检查数据和恢复/重起信息，还包括用户传送的必要的相关数据。

在分布式的工作流执行服务中，一般由多个工作流机协同工作来推进工作流实例的执行。每一个工作流机负责控制和管理一个过程中的一部分活动，使用相关的资源和应用工具来完成这些活动的执行。不同的工作流机之间进行分工和协作，在一个工作流机完成了其负责的一部分活动后，需要将相应的完成信息传递给其它工作流机。这些不同的工作流机之间的协作需要良好的底层支持系统（如可靠的消息系统）的支持。首先这种执行服务方式需要有共同的命名方式和管理范围，便于维持过程定义和用户/应用名称的一致性。分布式的工作流系统采用特定的协议来同步各工作流机，并传递相应的控制信息。不同的工作流执行服务中定义的协议通常是不同的，它们大多数都是厂家自己定义的。在选用不同的工作流系统产品进行协同工作时，需要在各工作流机之间提供一个标准的接口或格式来进行这些不同协议的转换。所提供的标准的接口和格式应包括以下几个方面的内容：

1) 一个共同的命名机制；

2) 支持共同的过程定义对象和属性；

3) 能够传递相关的工作流相关数据，并控制过程实例的生成；

4) 能够在异构的工作流机间传递过程、子过程及活动信息；

5) 支持共同的管理职能。

有关分布式工作流执行服务的实施和执行是一个比较复杂的过程，除了上面说明的需要有共同的协议与标准接口外，在具体的软件实现上还有很多细致的实施问题需要解决。本书的后续几章中将陆续对分布式工作流执行服务的设计与实施问题进行介绍。

工作流机是一个为工作流实例的执行提供运行服务环境的软件或“引擎”。它是工作流执行服务的核心，是执行企业经营过程的“业务操作系统”的内核。从提供的功能上看，它主要完成以下任务：

1) 对过程定义进行解释；

2) 控制过程实例的创建、激活、挂起、终止等；

3) 控制活动实例间的转换，包括串行或并行的操作、工作流相关数据的解释等；

4) 提供支持用户操作的接口；

5) 维护工作流控制数据和工作流相关数据，在应用或用户间传递工作流相关数据；

6) 提供用于激活外部应用程序和访问工作流相关数据的接口；

7) 提供控制、管理和监督工作流过程实例执行情况的功能。

工作流机的一个重要功能就是控制过程实例和活动实例的状态转换。工作流管理联盟提出的参考模型中对过程实例运行状态和活动实例状态进行了定义，并给出了状态转换的条件。图3.5和图3.6分别描述了过程实例和活动实例各个状态之间的转换。

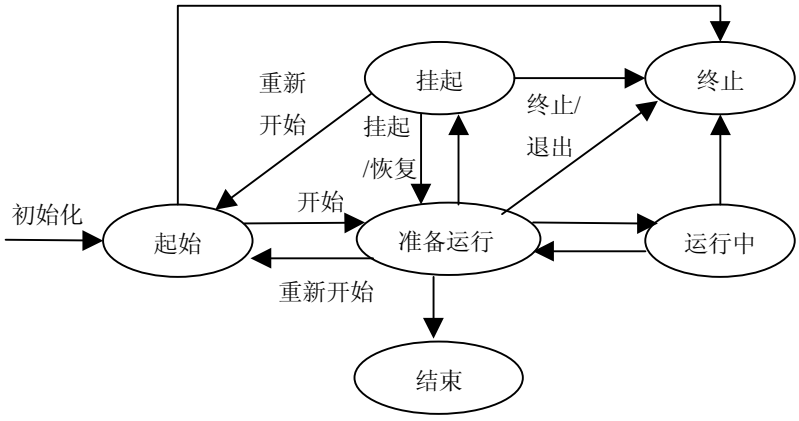


图3.5 过程实例状态转换图

过程实例包括以下几种运行状态：

1) 起始（initiated）：一个过程实例已经生成，但该过程实例目前还没有满足开始执行的条件；

2) 准备运行（running）：该过程实例已经可以执行，但是还不满足开始执行第一个活动并生成一个任务项的条件；

3) 运行中（active）：一个或多个活动已经开始执行（也就是已经生成一个或者多个任务项并分配给了合适的活动实例）；

4) 挂起（suspended）：该过程实例正在运行，但处于静止状态，除非有一个“重启”的命令或者外部事件促使该过程实例回到准备运行状态，否则所有的活动都不会执行；

5) 结束（completed）：该过程实例执行已经完成，并且满足了结束该实例的条件，工作流管理系统将执行过程实例结束后的操作（如统计），并删除该过程实例；

6) 终止（terminated）：该过程实例在正常结束前被迫终止（如出现错误或者异常情况），工作流管理系统将执行补救措施，并删除该过程实例。

图3.6给出了活动实例的运行状态：

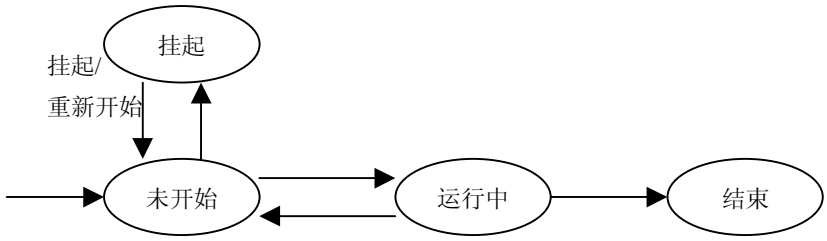


图3.6 活动实例状态转换图

1) 未开始（inactive）：该活动实例已经生成但还没有被激活（例如活动开始条件没有满足）；

2) 运行中（active）：该活动实例已经被激活，正在进行运行；

3) 挂起（suspended）：由于出现异常情况或者错误，该活动实例被暂时挂起，活动处于静止状态；

4) 结束（completed）：该活动已经执行完毕，工作流管理系统将进行活动结束后的导航工作，激活下一个符合启动条件的活动实例。

3.5 客户端功能

1. 工作流客户应用

工作流管理系统的客户端功能是指用户操作工作流管理系统分配的任务或者活动的功能。它由工作流任务表管理器和用户操作共同完成。工作流任务表是指分配给一个特定用户（或一组用户）处理的由任务项组成的队列。工作流任务表管理器是一个软件模块，它负责管理工作流任务表，并完成与最终用户的操作进行交互。工作流任务表管理器可以作

为工作流管理系统的一部分提供给用户，它也可以是用户自己编写的程序。例如用户自己设计开发了一个通用的任务表管理器，它可以实现与多个工作流产品的集成，从而提供通用的用户交互界面。

在实际应用中，经常需要将工作流管理系统集成到用户的其它桌面应用（如办公自动化系统、电子邮件系统）环境中，从而为最终用户提供一个集成化的统一任务管理系统。在一般应用环境中，计算机操作系统的异构性是比较普遍的情况，这就对工作流执行服务与工作流客户端应用之间的通信机制提出了较高的要求，如要求通信机制具有比较好的柔性能够适应不同的操作系统环境。

在工作流模型中，客户端应用与工作流机的交互通过定义良好的接口完成。这个接口就是工作流任务表。在最简单的情况下，工作流机通过存取工作流任务表来完成特定任务到特定用户的分发过程，而工作流任务管理器存取工作流任务表是为了获取任务项，将它们提供给用户进行处理，并得到处理结果。

通过任务表管理器来进行任务处理的方式适用于需要人员参与的活动。这种情况下，工作流机通过任务表管理器对活动进行控制。工作流管理联盟提供了四种可能的通过任务表来实现工作流客户与工作流机之间的通讯方式，如图3.7所示。其中第一种适合用于支持集中式的工作流管理系统结构，另外三种适合于分布式工作流管理系统结构。

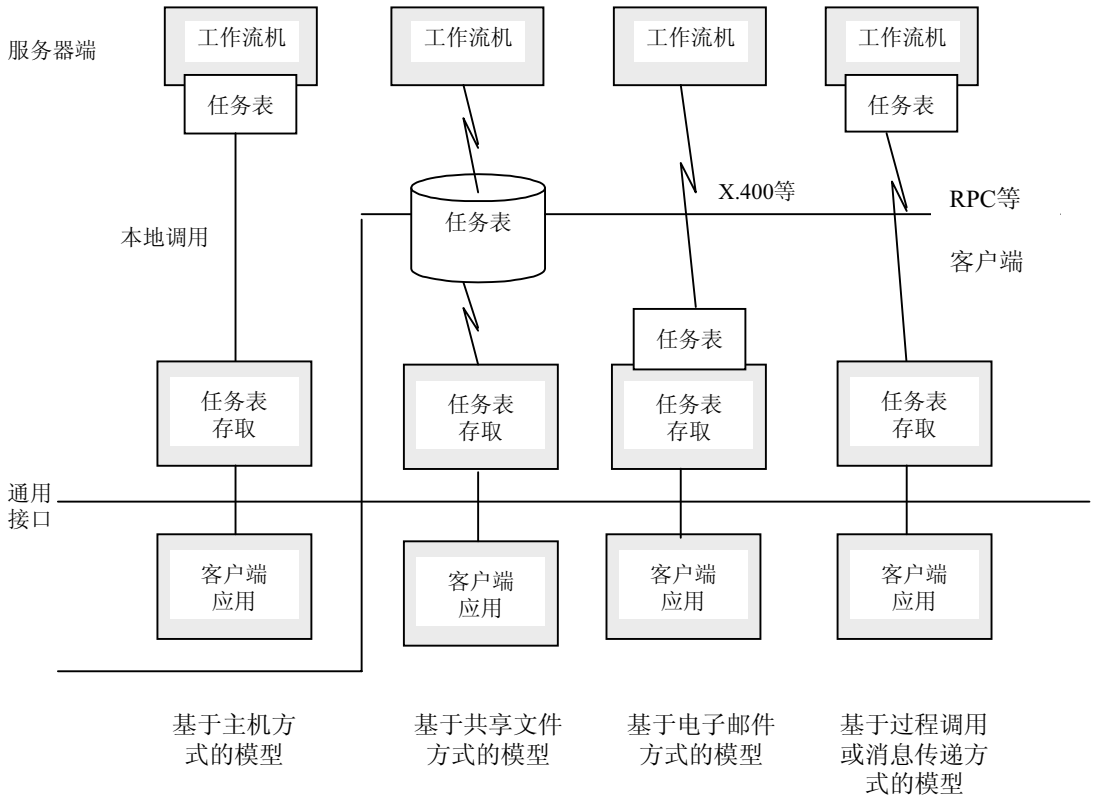


图3.7 四种任务表管理器的实现方法

1) 基于主机方式的模型（Host Based Model）：这种方式适合于集中的情况。此时，客户端应用程序、任务表管理器、任务表和工作流机都在中央的主服务器上，用户通过终端来获得任务表。

2) 共享的文件库模型（Shared Filestore Model）：在这种情形中，客户应用程序和任务表管理器位于用户的工作站上，而工作流机位于中央服务器上。任务表位于一个客户应用和工作流机都能够访问的共享文件系统中。

3) 电子邮件模型（Electronic Mail Model）：这里，客户应用和任务表管理器位于用户的工作站上，工作流机位于中央主机上。所有的通讯都使用电子邮件。此时，任务表一般位于客户端。

4) 过程调用或消息传递模型（Procedure Call or Message Passing Model）：客户应用程序和任务表管理器位于用户的工作站上，任务表和工作流机位于服务器端。用户通过RPC（远程过程调用）或者其它的消息传递机制来获得任务表。

激活工作流任务表中任务项的过程可以由工作流客户应用或最终用户来进行控制，激活的过程涉及启动应用程序、连接工作流相关数据等操作。为了实现新的任务项到工作流任务表的加入、已经完成执行的任务项删除、未完成活动的挂起等操作，WfMC定义了一系列过程规则来实现工作流执行服务和客户端应用程序之间的协作。有关它们的详细介绍在本节的下一部分（工作流客户应用接口）中给出。

在工作流任务管理器激活某些应用时，必须提供一些与任务项相关的信息，如应用的名称、地址等信息。这些信息可以采用不同的方式得到，如可以直接存放在工作流任务表中，还可以通过工作流机和工作流任务表管理器之间提供的交互接口来传递。在后一种情况下，工作流客户应用还可以通过编制一个接口函数通过直接调用的应用（接口3）来获取必要的信息。

在实际的应用环境中，工作流管理系统在设计和实施中必须提供足够的柔性以满足不同的应用需求和不同的用户操作方式存在的差别，如在实际应用中，一个工作流任务表可能包含一个过程实例产生的多个任务项，也可能包含多个过程实例产生的多个任务项。一个工作流任务表管理器可能与多个不同的工作流机和工作流执行服务进行交互，这些不同工作流机和工作流执行服务可能采用的是完全不同的实施方式（采用不同的编程语言、运行在不同的硬件平台上）。因此，连接工作流机与工作流客户应用之间的接口定义必须具有足够的柔性，它能够为实现不同工作流机和应用程序之间交互提供支持，这些工作流机

与应用程序可能使用不同参数、术语和不同的数据结构表示方式。这些不同表示方式反映在以下具体的形式上：

- 过程和活动标识符；
- 资源名称和地址；
- 数据引用和数据结构；
- 通信机制。

2．工作流客户应用接口

解决上面提到不同表示方式问题的方法之一是定义一组标准化的 API 来屏蔽这些不同的工作流机和应用实施中出现的差别。在与某个具体的工作流系统进行连接时，使用标准化的 API 接口中的函数实现应用参数到工作流系统的映射，通过不同的通信机制来实现参数的传递和连接。以一个使用电子邮件为通信手段的工作流系统为例，工作流任务管理器可以使用普通的本地邮件箱来接收到达的邮件，而不是去开发一个专用的邮件接收箱。在这种情况下，工作流任务管理器需要采用合适的方法来过滤掉所有非工作流系统产生的电子邮件项。同样采用类似的方法，工作流任务管理器可以将工作流应用产生的电子邮件添加到本地的电子邮件发送箱中。在这种情况下，一个简单的互操作场景就可以通过使用标准的电子邮件交换格式实现。

图 3.8 给出了客户端应用的交换接口形式。

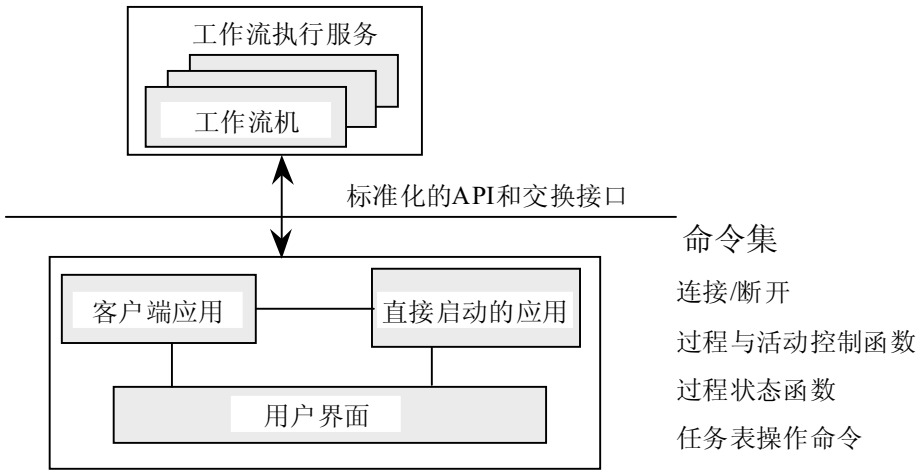


图 3.8 客户端应用接口

客户端应用接口提供的一些基本操作包括：建立连接、工作流定义、过程控制、过程状态查询、任务表/任务项处理、过程监控、数据处理与管理。这些操作的具体含义是：

1) 建立连接：建立/断开参与工作流管理的系统；

2) 工作流定义：获取/查询（可以使用一定的查询规则）工作流过程定义的名称和属性；

3) 过程控制：包括了一组创建、控制、查询过程实例的函数。如：

- 创建/启动/终止每个过程实例；
- 挂起/恢复过程实例的执行；
- 强制某个过程实例或活动改变其状态；
- 赋值/查询一个过程实例或活动的某个属性，比如优先级。

4) 过程状态查询：完成对过程实例状态的查询，包括：

- 打开/关闭对过程实例的某个查询命令，设置可选的过滤（查询）规则；
- 按照选定的过滤规则获取过程或活动实例的详细信息；
- 获取某个特定过程或活动实例的详细信息；

5) 任务表/任务项处理：完成对任务表和任务项的查询处理，包括：

- 打开/关闭对任务表的查询操作，设置可选的过滤（查询）规则；
- 按照选定的过滤规则获取任务项信息；
- 发布一个特定任务项的选择、分配和完成信息；
- 赋值/查询一个任务项的属性；

6) 过程监控：过程监控函数是具有特定权限的人员才能够操作的函数，它们一般不向所有的客户端应用和用户开放，这些函数包括：

- 改变工作流过程定义和它的尚存的实例的运行状态；
- 改变一个特定类型的所有过程和活动实例的状态；
- 给一个特定过程和活动所有实例的属性进行赋值；
- 终止所有的过程实例的执行；

7) 数据处理：获取/返回工作流相关的或应用的数据；

8) 管理：支持在工作流应用编程接口基础上实现进一步的管理功能；

上面给出的这些函数为实现由工作流任务管理器启动客户端应用提供了良好的基础，并为存取过程、活动、任务项的属性以及工作流相关的数据提供了操作手段。

3．直接启动应用

这种情况适合于不需要人员参与的活动。在简单的情况下，工作流机通过过程模型中定义的活动信息、应用程序类型和所需的数据来激活应用程序。被激活的应用程序可以与工作流机位于一台计算机上，可以位于相同的运行平台上，也可以位于通过网络可以访问

的不同计算机平台上。模型定义为应用启动提供了有关应用程序的类型、地址等的足够信息，因而允许工作流机激活该程序并执行相应的动作。

一般情况下，企业的应用系统具有分布性和异构性，大部分应用分布在不同硬件环境下，采用了不同软件平台开发环境和运行环境，更为关键的是这些应用是不同的软件厂商提供的产品，它们在接口参数、相关数据结构和通信机制上存在非常大的区别，工作流管理系统不可能提供对所有应用的直接启动功能。

但是也有许多应用是可以紧密地集成到工作流管理系统中，它们完全由工作流机来进行启动并控制其执行，如文字处理系统。在更广泛的应用场合，采用应用代理的方式来启动应用是一种具有较好柔性的方法。图3.9给出了直接启动应用的接口。

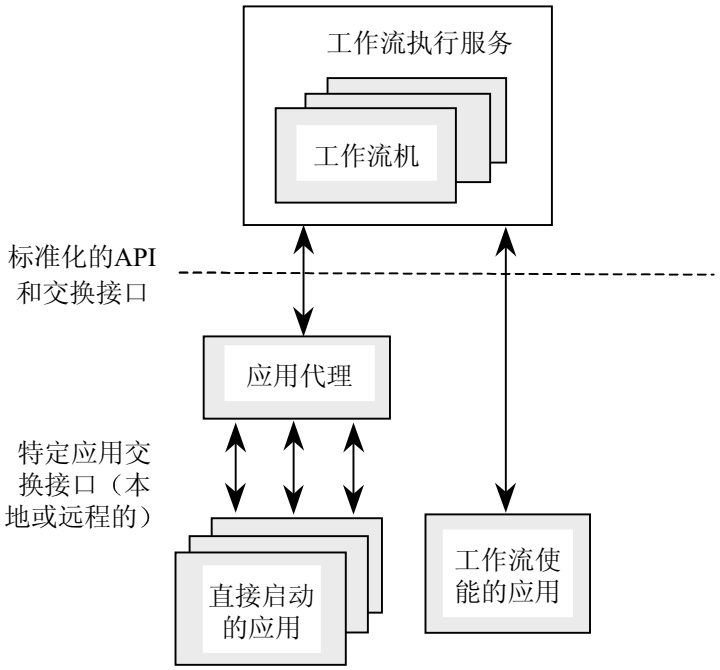


图3.9 直接启动应用接口

通过应用代理来进行直接应用启动为工作流管理系统的开发、应用系统的集成都带来很大的方便。应用代理与工作流机之间的数据交互和消息传递可以采用标准化的API和数据格式完成。而应用代理与直接启动的应用之间的数据交互和消息传递通过定义与开发面向特定应用的专用集成接口来完成。当用户的应用系统发生变化，或者需要自己开发新的应用时，无需改变工作流管理系统的结构与操作模式，也无需修改应用代理与工作流机之间的接口，用户仅需要修改应用代理与这个特定应用之间的接口。因此，采用应用代理是提高工作流管理系统柔性和适应性的有效方法。

3.6 工作流执行服务之间的互操作性

工作流管理联盟的目标之一是定义一个标准，使得不同厂商提供的工作流产品能够协同工作，整个系统能够无缝地在各个产品之间传递任务项，实现应用集成。工作流管理联盟在互操作性上的工作集中在提供一系列互操作的场景，从简单的任务传递到传输整个工作流过程模型和工作流参考数据。尽管也可以设计与规定很复杂情形下的工作流互操作规范，如不同厂商提供的工作流机协作实现复杂的工作流执行服务，但是这在目前还不可能实现，因为它要求所有的工作流机都能够解释同一个过程模型，共享一套工作流控制数据，并在异构的工作流机环境下共享过程实例的状态。就目前比较现实的情况来说，比较切合实际的目标是在不同的工作流执行服务间传递过程的部分内容，并支持其实例的运行。

3.6.1 互操作模型

在异构的工作流系统间需要传输的信息流和控制流有：活动或子过程的激活；过程/活动实例的状态；工作流相关数据与应用数据的传输；同步点协作；过程模型的读写等。工作流管理联盟提出了四种可能的互操作模型，它们覆盖了互操作的不同能力层次。

1. 链状服务模型

模型1是链状服务模型。这种模型允许过程实例1中的一个点与过程实例2中的一个点相连。连接的两点可以是过程实例中的任意两个点，而不仅仅限于开始和结束点，如图3.10所示。

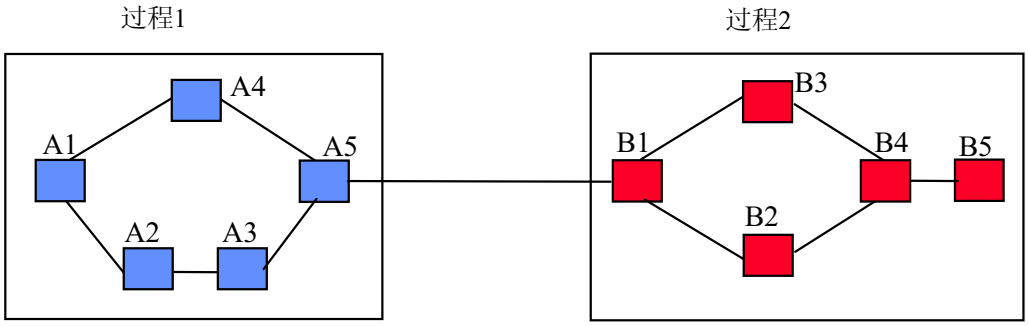


图3.10 链状服务互操作模型

这种模型支持在两种不同的工作流环境中传递单个任务项（一个过程实例或活动实例），然后由接收方单独执行，双方无须再同步。在实施过程中，可以通过一个网关应用程序来实现数据格式转换、过程和活动名称的映射等等，或者可以通过工作流服务的方式来实现这种互操作模式，如在两个工作流服务间调用标准的API（应用编程接口）函数。

2. 网状的子过程

模型2是网状的子过程模型。在这种模型中，一个特定工作流服务域中运行的一个工作流过程实例可以做为另一个不同工作流服务域中的一个活动（也就是子过程），如图3.11所示。同模型1中的情景一样，数据的传输可以通过应用程序网关函数或者在两个工作流服务之间调用API（应用编程接口）来实现。

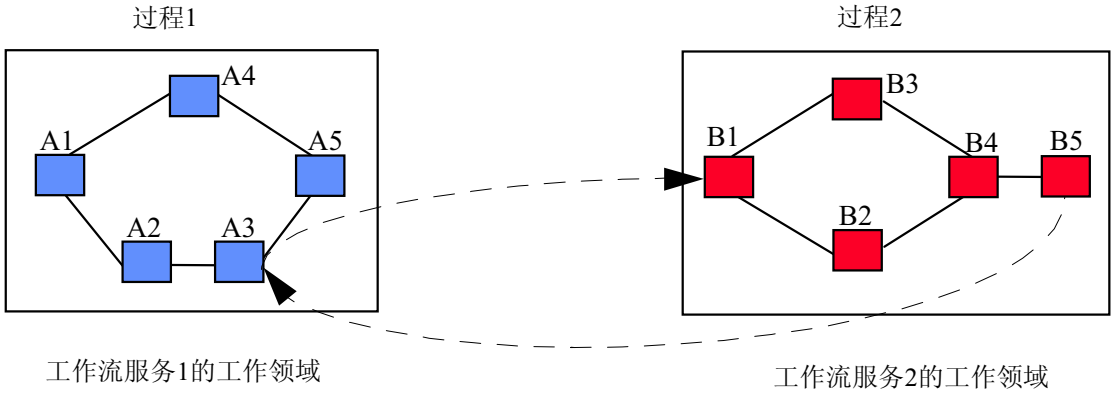


图3.11 网状的子过程互操作模型

4. 端对端模型

模型3是端对端模型。假设在过程实例1中，共有C1到C6六个活动。其中C1、C2和C5在工作流机1上执行，C3、C4和C6在工作流机2上执行。这种情景要求双方都支持公共的通讯API（应用编程接口）规定，从而双方都能够解释一个公共的过程模型。在两个异构的工作流机间还需要传递工作流相关数据和应用数据。如图3.12所示。

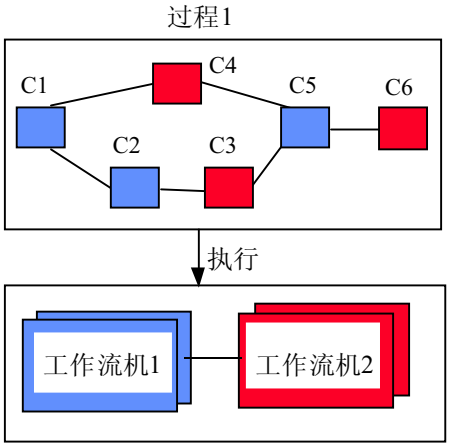


图3.12 端对端互操作模型

我们现在只是简单地考虑了在过程建模时就定义好由哪个工作流机来完成哪项活动作者情况。当一个活动可以由任意一个工作流机执行，或者任意一个工作流机都可以开始和

终止该活动实例时，情况就变得十分复杂。为了能够实现这种复杂情况下的多工作流机的互操作和协同工作问题。必须定义一套完整的机制来负责解决多工作流服务协作中的系统管理、安全性和恢复问题。

4. 并行同步模型

模型4是并行同步模型。这种模型允许两个过程实例互相独立地在各自的执行环境中运行，但是要求在两个过程中存在同步点。当它们各自运行到同步点时，就会触发某个事件。这种机制可以用来实现并行执行线程间的调度、恢复数据的检查或者在不同过程实例中传递工作流相关数据等功能。如图3.13所示。

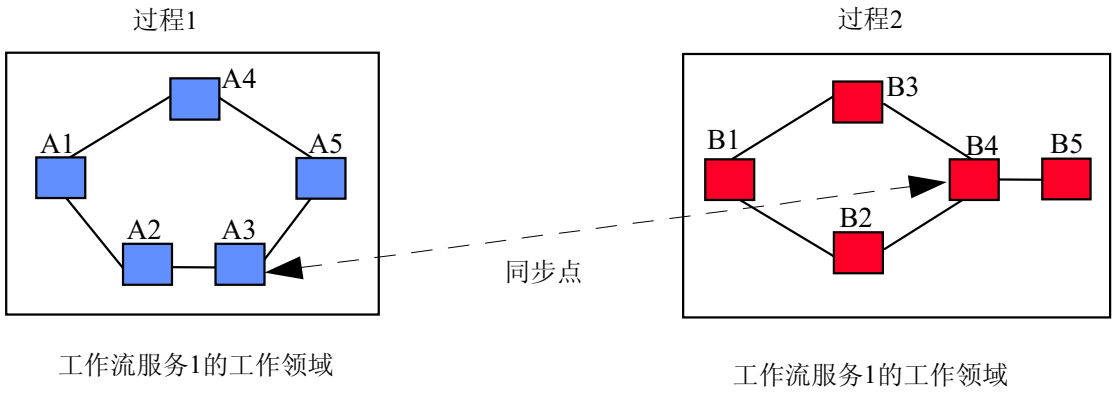


图3.13 并行同步互操作模型

3.6.2 两种互操作情况

图 3.14 给出了异构工作流系统之间交互的信息与控制流情况。它反映的是一般情况下在异构工作流系统之间交互的信息，WfMC 将异构工作流之间交互信息的接口定义为接口 4。在不同工作流管理系统之间实现互操作可以采用不同的技术来实现，如

1) 直接编制 WFMS 之间的互操作函数来实现交互；

2) 在 WFMS 之间进行消息传递；

3) 使用公共网关来连接不同的 WFMS 的协议和格式；

4) 使用共享数据存储方式来实现信息交互。

前面我们已经指出，虽然 WfMC 的目标是实现不同工作流产品之间的连接和互操作，但是由于商业和技术实现上的原因，不同的工作流管理系统之间是难以实现完全的互操作的。因此，WfMC 在接口 4 上定义了 8 个不同的互操作层次，按照互操作程度由低到高，这 8 个层次分别是：

- 层次 1－－没有互操作；
- 层次 2－－共存：在这种情况下，由工作流系统和用户自己去解决工作流管理系统之间的互操作
- 层次 3－－唯一公共网关：在工作流机之间导航活动，传递工作流与应用数据；
- 层次 3a－－公共网关 API：通过一组标准化的指令集来简化网关的交互；
- 层次 4－－有限通用 API 子集：基于共享的 API 类实现工作流机之间的直接交互；
- 层次 5－－完全通用 API 集：所有的 WFMS 的操作都可以被另外一个系统访问；
- 层次 6－－共享定义格式：不同的工作流管理系统可以基于同一个过程定义执行实例；
- 层次 7－－协议相容：允许工作流管理系统无缝的交换过程定义、工作流与控制数据；
- 层次 8－－相同视觉与接触能力：定义完全统一的标准化用户界面。

但是，一个非常难以回答的问题是：在某个应用企业中，那个层次的互操作能够满足或者最符合企业组织过程的需求？

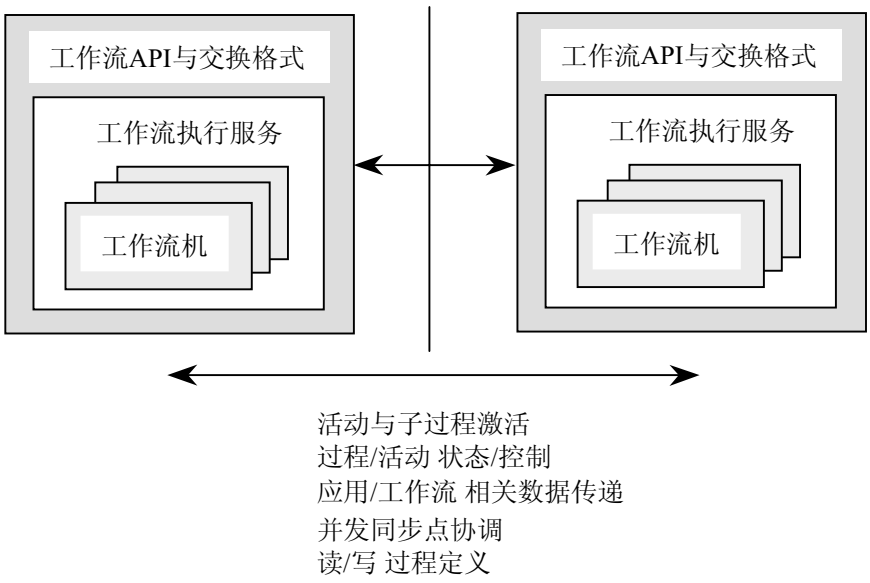


图 3.14 工作流互操作接口

在这里我们主要考虑工作流管理系统的互操作和协同工作领域的以下两种互操作情况：

1) 能够对公共过程模型进行解释，即能够共享过程模型定义；
2) 在运行时，能够在不同的执行服务中传递不同类型的控制信息、工作流相关数据或应用数据。

下面对这两种互操作进行详细的解释。

1. 在不同的领域内共享过程模型定义

如果两个工作流执行服务都能够对一个过程模型给出正确的解释（例如对一个由公共建模工具生成的模型），那么这两个运行环境就可以共享过程定义中的对象和它们的属性，如活动、应用、组织和角色名称、导航条件等。这使得工作流机可以很容易地将任务或子过程传输给在一个公共的命名机制和对象模型环境下运行的其它工作流机。这种互操作与上面介绍的端对端互操作模型的情况十分类似。

如果由于实际应用情况的限制，不可能建立这种共享过程模型的情况下，还可以采取在运行过程中传输过程模型部分细节的方法来实现这种互操作。过程建模接口所定义的 API 中提供了允许工作流执行服务获取有关对象和属性数据的函数，这使得工作流机可以在运行过程中获得分配给它执行的活动或子过程相关的过程模型信息。

当以上两种方法都不可能实现互操作时，互操作就只能使用网关来实现。在这种方法中，两种运行环境通过应用程序互操作网关（application interworking gateway）来映射对象名称和属性。最简单的情况是两种环境分别使用自己的建模工具和模型格式，两种环境之间的映射完全由网关负责提供转换接口来实现。

2. 运行时交互控制信息

在运行过程中，可以通过 WAPI（工作流应用编程接口）来调用在工作流执行服务间传递的控制信息，实现在指定的执行服务上完成子过程或单个活动的功能。当两种运行环境都支持共同的 WAPI（工作流应用编程接口）调用和过程模型定义（包括命名机制、工作流相关数据和工作流控制数据）时，可以在工作流机之间直接实现该功能，但这种做法要求双方都支持 WAPI（工作流应用编程接口）基本元素的公共协议。

当不满足上述情况时，可以使用 WAPI（工作流应用编程接口）来调用在两种执行环境中提供的网关服务实现互操作。该服务负责映射两种不同环境下的对象和数据视图，并在必要的时候支持不同的工作流服务协议。

图 3.15 描述了通过 WAPI（工作流应用编程接口）实现基于网关的互操作情形。由图中可以看出在领域 1 中的一个独立活动有可能对应于领域 2 中的一个活动或者一个子过程。

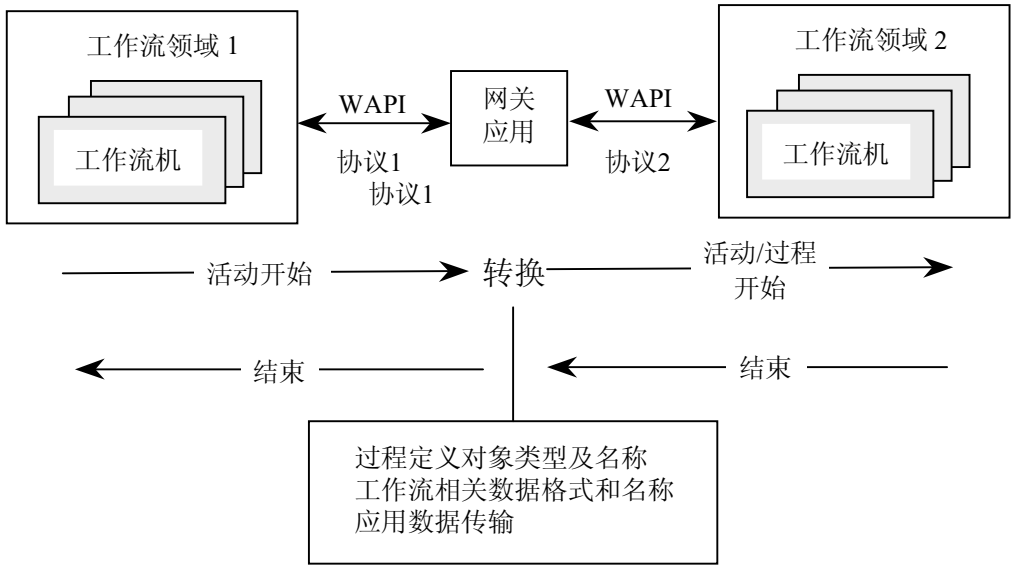


图 3.15 通过网关实现互操作

3.7 系统管理和监控工具

系统管理与监控是复杂软件系统一个非常重要的功能。在实际应用，尤其是在企业应用中，系统在运行过程中不可避免地会出现许多意外情况，正确及时地处理这些意外情况对于保证应用系统良好运行具有十分重要的意义。除了在系统设计实施中充分考虑各种情况，提高应用系统的可靠性和鲁棒性外，提供良好的系统管理与监控工具对于提高系统的应变能力，充分发挥用户处理复杂问题的聪明才智是一个有效的方法。

工作流系统管理与监控工具能够对工作流在整个组织内的流动状况进行监视，并提供一系列的管理功能，实现安全性、过程控制和授权操作等方面的管理。它包括以下几个方面的功能：

1) 建立、设置和优化组成工作流管理系统的各个软件；

2) 对过程模型进行实例化；

3) 将过程模型中的角色实例化；

4) 将运行中的过程实例、活动实例和数据分发到各个工作流机中；

5) 启动、挂起、恢复和终止过程实例；

6) 管理正在执行的过程实例，并对正常结束或异常退出的过程实例的历史数据进行统计和分析。

图 3.16 给出了工作流系统管理与监控的接口结构。

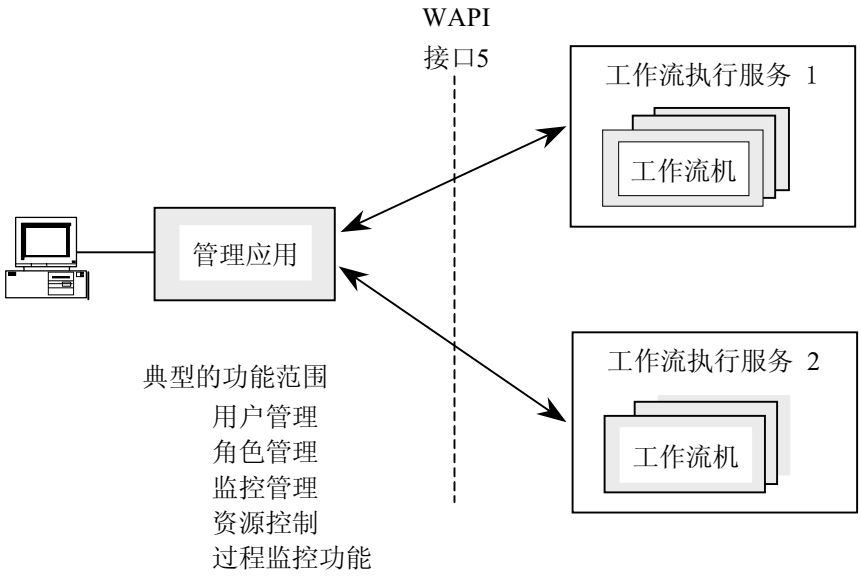


图 3.16 工作流系统管理与监控接口结构

3.8 WAPI 与接口

WAPI 是一组工作流应用编程接口函数及其相应的数据交换格式，通过调用这些接口函数可以完成工作流联盟定义的 5 类接口的功能。这 5 类接口需要的操作函数包括一组 API 调用函数和一些数据交换函数。它们是：

1) API 调用：完成连接建立、过程控制、活动管理等功能，包含以下类型的功能函数供编制接口程序使用：

- 建立连接；
- 操作工作流模型及其对象；
- 过程控制；
- 过程监控；
- 过程状态检测；
- 活动管理；
- 数据处理操作；
- 工作流任务表/任务项处理操作；
- 用户管理；
- 角色管理；
- 监控管理；
- 资源控制。

2) 数据交换函数：数据交换格式的定义与转换函数，能够满足过程数据传递和工作流相关数据传递的需要。

API 调用通常可以以它们的逻辑功能、操作的数据类型（调用参数）及引用的数据结构等参数项进行定义。这些 API 定义需要与具体的程序实现语言（如 C， C＋＋）进行绑定，对于采用 COBRA 规范的接口可以使用 IDL（接口定义语言）方式进行映射。

WfMC 定义了 5 类接口，这些接口在本章的相应部分已经进行了一些介绍，为了给读者提供一个完整的工作流管理系统的接口描述，以下我们对 5 类接口的功能进行介绍。WfMC 定义的 5 类接口（图 3.2）是：

1) 接口 1，工作流服务和工作流建模工具间接口，包括工作流模型的解释和读写操作；

2) 接口 2，工作流服务和客户应用之间的接口，这是最主要的接口规范，它约定所有客户方应用与工作流服务之间的功能操作方式；

3) 接口 3，工作流机和直接调用的应用程序之间的直接接口；

4) 接口 4，工作流管理系统之间的互操作接口；

5) 接口 5，工作流服务和工作流管理工具之间的接口。

接口1：过程定义输入输出接口

这个接口为在不同物理或电子介质之间传递过程定义的信息提供了交互的形式和API调用函数。使用标准定义接口具有很多好处。首先，它实现了建模环境和运行环境的分离，使用某种建模工具创建的模型可以运行在不同工作流产品上。其次，它使得多个工作流产品可以协同工作，构成一个工作流执行服务，并运行根据同一个过程模型所生成的过程实例。工作流管理联盟提供的API函数在建模方面主要覆盖了以下几个功能：

1. 通信建立：各个参与的系统之间通讯的建立与断开；
2. 工作流模型操作：过程模型名称检索；对过程模型对象的读/写等；
3. 工作流模型对象操作：在建模工具中创建、检索和删除对象；创建、设置和删除对象的属性。

接口2：客户端函数接口

接口2主要定义了以下几个方面的功能：

1) 通讯建立：各个参与运行的系统之间通讯建立与断开；

2) 工作流定义操作（对过程模型定义操作）：对于过程模型的名称和属性的查询和检索功能；

3) 过程实例管理功能：创建/开始/结束一个过程实例；挂起/重新激活一个过程实例；强行改变过程实例或活动实例的状态；对于过程实例或活动实例的属性的查询；

4) 过程状态管理功能：设置过滤条件，打开/关闭对一个过程实例或活动实例的查询；设置过滤条件，获得部分过程实例或活动实例所需的详细信息；获得过程实例或活动实例的详细信息

5) 任务项列表/任务项处理功能：设置可选的过滤条件，打开/关闭一个任务项列表查询；设置过滤条件，获得任务项列表中的项目；对任务项的选择/重新分配/完成等消息的通知；设置和查询一个任务项属性

6) 数据处理过程：查询/返回工作流相关数据、工作流应用数据

7) 过程监控功能：改变一个过程模型和其现存的实例的运行状态；改变某种特定类型的过程实例或活动实例的状态；改变某种特定类型的过程实例或活动实例的属性；结束所有的过程实例

8) 管理功能：其他的管理功能

9) 应用程序激活。

接口3：激活应用程序接口

激活应用程序的API（应用编程接口）函数应覆盖以下几个方面的功能：

1) 通信建立：与应用程序（或应用程序代理）通信的建立/断开

2) 活动管理功能：活动开始；挂起/重新激活/退出活动（需要一个非同步的应用程序界面）；活动结束通知；查询活动属性；消息事件（如同步信息等）

3) 数据处理功能：提供工作流相关数据（在活动执行前向应用程序提供数据，活动结束后从应用程序中获得所需数据）；提供工作流应用数据或提供数据地址

更复杂的情况（如异构的工作流机之间的协作）则需要在不同的工作流机之间传递激活应用程序所需的数据：或者在运行时传递，或者在建模过程后传递模型的部分信息。

接口4：工作流执行服务之间的互操作接口

工作流执行服务之间需要提供 WAPI（工作流应用编程接口）来实现互操作，无论是在两个工作流执行服务间的直接调用还是通过网关函数。前面讨论的许多 WAPI（工作流应用编程接口）函数都适用于工作流互操作的情形：

1) 连接的建立；

2) 对工作流模型和其中对象的操作；

3) 过程实例控制和状态函数；

4) 活动管理函数；

5) 数据处理函数。

接口5：系统管理与监控接口

工作流管理联盟提供的第五类接口主要是为了实现对工作流的管理和监视。WAPI（工作流应用编程接口）包括以下几个方面的内容：

1) 资源控制：设置/取消/修改过程或活动实例的并发层次（concurrency levels)；

2) 角色管理：定义、删除和修改角色－参与者（role:participant）的关系；修改角色属性；

3) 用户管理：对用户或工作组的权限的建立、删除、暂停和修改；

4) 过程实例的管理：对过程模型进行实例化；修改过程实例或活动实例的状态；终止过程实例；

5) 状态管理：查询过程或活动实例的运行状态；

6) 审核（audit）管理：查询/打印/删除/启动新的统计过程轨迹（trail）或日志。

第 4 章 工作流技术研究发展情况

4.1 概述

工作流管理技术，在其发展的初期主要是由工作流产品开发的公司推动着其发展，随着它在实际应用中取得的良好效果而得到了充分的重视，并且得到了迅速的发展。相对于工作流产品市场的繁荣，工作流相关理论研究则显得有些滞后。在过去很长一段时间里，有关工作流方面的研究主要是由商品化的工作流管理系统的开发商所领导。本着把工作流产品推向市场的目的，这些开发商大多把研究的注意力放在了工作流管理系统的开发实施方面。目前，在工作流设计方法学、工作流概念模型等方面还没有形成一套比较成熟的理论和方法。

在工作流理论与实施技术方面，研究的主要内容包括：（1）工作流管理系统体系结构的研究；（2）工作流模型与工作流定义语言研究；（3）工作流的事务特性：研究如何实现高级事务处理技术与工作流管理技术的结合，用定义良好的模型语义与恢复机制来提高工作流系统的正确性与可靠性，从而能够更好地支持企业复杂的业务过程；（4）工作流实现技术：包括面向对象技术、异构分布式计算技术、图形化用户界面、消息通讯、数据库、WWW 等在内的与工作流系统的设计实现有关的各项技术及方法；（5）工作流的仿真与分析方法；（6）基于工作流的应用集成与互操作技术：研究异构应用系统的集成以及不同工作流系统之间的互操作问题；（7）工作流与经营过程重组（BPR）：研究如何通过工作流系统的实施来支持企业快速高效地实现经营过程重组；(8) 工作流技术的其它应用：研究如何将工作流技术在不同的领域进行应用，包括在 CIMS 中的应用。

上述主要研究问题可以分为三个方面：第一方面是工作流的理论基础，包括工作流管理系统的体系、模型与定义语言（工作流的建模方法、工作流模型的形式化表示、工作流定义语言）等的研究。这一部分工作目前相对来说比较薄弱，还有许多问题需要进一步研究。第二方面是工作流的实现技术，包括工作流的事务特性、各种先进软件技术的应用、工作流仿真。这方面研究工作的目的是提高工作流管理系统的性能，尤其是提高工作流管理系统的可靠性及其在处理大规模复杂的且具有并行业务的流程方面的能力。第三方面是工作流技术的应用，包括工作流实施技术、在不同应用领域的应用（如在企业经营过程重组、并行工程、敏捷制造）方法、应用软件集成等。这方面研究的目的是发挥工作流管理系统的优势，为解决具体应用领域内的问题提供有效实现手段。

工作流是一个年轻却又具有良好发展前景的研究方向。它源于计算机软件的商业应用，是一个多学科交叉的新领域，涉及计算机科学与管理科学的多种原则、方法与技术，包括数据库、形式化描述语言、应用与系统的集成、软件工程、文档管理、组织与资源管理、仿真技术、企业重组、分工协作等。工作流技术并非强调业务过程的全面自动化和完全不需要人的参与（这一点与 CIMS 的实施指导思想完全相同），而是把重点放在对工作流中任务状态的自动跟踪与监控，并对任务执行的条件和任务间的信息流进行明确定义，从而实现业务过程的高效运转，并能够在最大程度上实现业务过程的重用。工作流管理实现了业务过程逻辑与嵌入在单个用户应用中任务逻辑之间的分离，这种分离使得两种逻辑可以各自独立修改，而且同一个任务逻辑也能够在不同的过程逻辑（经营过程）中实现重用。

目前工作流技术的研究正日益受到人们的重视，许多大学和研究机构都开展了很多研究项目，取得了众多的研究成果，它们对工作流技术的发展做出了贡献。这些研究成果涉及上面我们提到的工作流管理的各个方面。在所取得的研究成果中，比较著名的有 IBM 公司 Almaden 研究中心研究开发的基于持久消息队列的分布式工作流管理系统－Exotica/FMQM、佐治亚大学计算机系研究开发的具有自适应能力的工作流管理系统-Meteor（Managing End-To-End OpeRations）、基于分布式主动数据库技术的工作流管理系统-WIDE（Workflow on Intelligent and Distributed database Environment）以及基于状态与活动图的工作流管理系统-Mentor（Middleware for Enterprise-wide Workflow Management）。其中，Exotica/FMQM 和 Meteor 是完全分布的工作流管理系统，它们都是由一个个具有自治能力的节点组成。每个节点都有一个节点管理器，节点管理器具有工作流机的功能，完成工作流机对活动的管理控制任务，如对活动完成情况的管理、活动间的导航、异常处理等。从层次上来说，每个节点都是平等的。WIDE 和 Mentor 则采用的是 C/S 结构，活动间的导航、活动执行情况的管理、异常情况的处理都是由位于服务器层的工作流机完成的。系统的各个节点之间存在着层次上的差别。

在本章随后的四节中，我们对这四个研究项目逐一进行介绍。在本章的最后一节，对工作流管理技术的其它一些主要研究成果及研究方向进行介绍。

4.2 基于持久消息队列的分布式系统-Exotica/FMQM

由于企业业务流程本身具有分布性，要求工作流管理系统能够支持这种分布式环境业务应用的集成运行。通常在基于客户/服务器方式的系统设计中，是把有关过程执行的信息

（工作流定义、工作流实例等）存放在网络上的某一个节点上，以便实现对过程的监控、活动执行情况的统计与活动的同步等操作。这种集中式的管理结构存在单点失败和性能瓶颈两个严重问题。单点失败是指在工作流管理系统的执行过程中，由于一个活动实例出现失败而造成整个工作流系统停止运行。对于一个大型企业，它可能同时在并行地运行成千上万个过程实例，出现异常故障和活动执行失败这种情况是不可避免的，问题是如何减少异常及失败情况对于企业经营过程的影响。由于出现一个活动的失败而导致整个工作流系统停止运行是非常糟糕的情况，所以工作流管理系统应该尽可能提高其可靠性，避免这种单点失败情况的发生。性能瓶颈问题是指大量的应用和数据都要访问一个集中式的服务器，这必然会导致网络和服务器成为提高整个系统性能的瓶颈。

IBM Almaden 研究中心开展的研究项目 Exotica 的主要目的就是要将先进的软件技术应用到工作流管理系统中，以期提高工作流管理系统在正确性和可靠性等方面的性能水平，尤其是在大规模企业应用情况下，系统的可靠性与正确性对于工作流管理系统能否发挥其效益具有决定性的作用。大规模的含义是指系统的运行涉及非常多的人、地点或者过程。以 IBM 的工作流管理系统 FlowMark 的应用为例，在一个实际应用场合，一个月中运行的过程数量超过 30 万个；第二个实际应用场合中，将超过 4000 个不同分布地点的应用连接起来运行；第三个实际应用场合，超过 10 万个不同的用户共同参与来完成一个任务。

Exotica/FMQM （FlowMark on Message Queue Manager）是 Exotica 提出的一种完全分布的工作流管理系统。它基于 IBM 的原有工作流管理系统 FlowMark，通过扩展其功能而实现的一个能够支持大规模复杂应用的高性能分布式工作流管理系统。整个工作流管理系统由许多具有自治能力的节点组成，每一个执行节点都是相互独立的，工作流过程的执行不以某一个节点为中心，实现了完全分布。节点之间通过持久消息（Persistent Messages）的方式来实现工作流执行情况等相关信息的传递，即某个节点的一个活动实例执行完成后，通过可靠的消息队列在节点之间传递活动已经完成了的相关信息。这种运行机制避免了在过程运行中不断与服务器通信所造成的瓶颈。而且，这种体系结构也可以提高系统的鲁棒性，避免了单点失败造成的严重影响。在 Exotica/FMQM 中，如果其中的一个节点由于故障而停止运行，其余的节点还能够正常运转，不会造成所有的活动都被迫终止。这种方式大大地提高了系统的可靠性、可扩展性以及柔性。

4.2.1 建模方法

Exotica 使用 IBM 的工作流产品 FlowMark 提供的建模工具进行模型定义。FlowMark 有一个集中式的数据库服务器管理所有的数据，数据库系统采用面向对象的数据库系统 ObjectStore。系统还有三个主要部件组成，它们分别是：

1) 客户端建模工具：用于建立工作流模型；

2) 客户端运行工具：用于完成工作流执行时的交互工作；

3) FlowMark 服务器：用于运行控制。

需要指出的是在一个工作流管理系统的实施中，可以安装许多个以上三种部件。FlowMark 服务器作为数据库服务器的客户甚至可以与数据库服务器不在同一个局域网上。

图 4.1 给出了使用 FlowMark 建立的一个简单的过程模型。

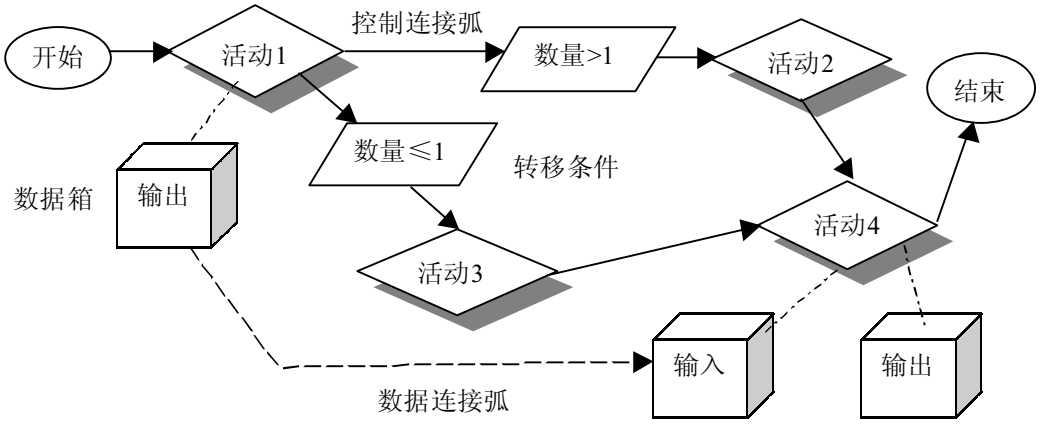


图 4.1 FlowMark 的基本过程模型

FlowMark 以工作流管理联盟提出的工作流基本模型为基础，所建立的模型由以下几个部分组成：

1）活动：活动是指组成业务过程的一个操作或一个功能单元，例如修改某个文档。每个活动都有名称、类型、开始条件、结束条件等属性。除此之外，每个活动还有一个输入数据箱和输出数据箱。FlowMark 支持两种类型的活动，人工完成的活动和由系统自动完成的活动。对于人工完成的活动，建模人员可以指定某个部门、某种角色类型或者某个具体的人员来完成该活动。对于系统自动完成的活动，建模工具提供了专门的界面让建模人员指定需要激活的应用程序，包括指定程序所在的节点、程序所处的具体目录、位置。当系统运行到该活动时，该应用程序会自动被调用。

2）控制流：控制流是指一个模型内所有的活动间的控制连接弧。每个控制连接弧和工作流

参考模型中的转换条件功能类似，主要用于描述活动间执行的顺序及转移的条件。例如在图 4.1 所示的过程模型中，在活动 1 结束后，如果活动输出的某个结果（比如说文档数目）数量大于 1 则执行活动 2，如果数量小于等于 1 则执行活动 3。

3）输入数据箱：输入数据箱为该活动激活应用程序提供必需的参数。

4）输出数据箱：输出数据箱用于存储应用程序的输出结果。它既可以是最终输出的结果，也可以作为下一个活动的输入参数。FlowMark 的建模工具提供了从输出数据箱到输入数据箱的映射机制。

5）数据流：数据流由过程模型中所有的数据连接弧组成。它描述了输入数据箱和输出数据箱之间的映射关系，使得活动间可以相互传递数据。如图 4.1 中从活动 1 的输出数据箱到活动 4 的输入数据箱之间的数据流动。输入/输出数据箱和数据流构成了参考模型中的工作流相关数据。

6）开始条件：开始条件描述了一个活动可以执行的条件，它对应于参考模型中活动的前条件。

7）结束条件：结束条件描述了一个活动可以结束的条件，对应于参考模型中活动的后条件。

8）除此之外，在过程模型中还有一个开始节点和终止节点，它们主要用于在整个系统中统一过程实例的开始和结束状态。

在 FlowMark 中活动之间的导航是由控制流和数据流来实现的，其中控制流决定了活动之间的相互执行顺序和转换的条件，而数据流则描述了一个过程实例中数据在不同活动间的流动。

Exotica/FMQM 面向分布环境下的用户，其过程实例的运行也分布在整个运行环境内。建模过程中，建模人员需要指定每一个活动实例具体的执行地点，以便将过程实例中的活动分配到相应的节点上执行。

4.2.2 实现技术

持久消息、过程绑定、过程实例运行是 Exotica 研究项目重点解决的关键技术问题。在 Exotica/FMQM 中，建模结束后首先要完成过程模型到各个执行节点的绑定，然后才是过程实例的运行。模型的绑定是对模型中的信息进行“编译”，将各个节点执行活动所需的模型信息发送到各个节点中，并为过程实例的运行做好准备，这个过程就是通常所说的实例化过程。过程实例的执行是由各个节点根据模型的定义和具体实例运行的情况来主动完

成实例的推进和运行，直至运行完毕。以下分别对这三个方面的实现技术进行讨论。

1．持久消息

Exotica/FMQM 的设计方案建立在底层消息传递系统之上，类似的产品有 DEC 的 MessageQ、Novell 的 Tuxedo/Q、IBM 的 MQSeries。这些消息系统提供了中间件的功能，它们为上层的应用屏蔽了复杂的通讯实现代码以及操作平台、网络协议的异构性，通过标准的 API 函数来提供各项消息服务。这些产品的特点很适合于用来连接分布式应用，实现工作流管理的功能。

在消息系统所构成的信息基础结构之上，分布的执行节点（即运行不同应用的网络节点）间通过发送与响应消息相互联系。持久消息则是指被保存在可恢复消息队列中的消息，它们一旦被放入消息队列，并且在相应的消息处理事务开始后，这条消息将一直被保存，直到处理完毕后，才从消息队列中取走。每一个执行节点都有一个（或多个）具有可恢复能力的消息队列，每个队列对应于一个过程实例。消息队列中的消息通过 PUT 与 GET 这样的函数调用来进行插入或获取。GET 函数调用可以获得队列中的任何一条消息，不仅仅限于位于队列首位的消息，这也是实现持久消息服务所要求的基本特性。另外，对于 PUT 与 GET 这样的操作，还应该具有事务特性，以保证在某些例外情况发生后，队列中的消息不会因此而丢失。

2. 过程模型的绑定

当建模人员完成一个过程模型的建立以后，系统将对模型进行“编译”，确定执行这个模型中活动的所有相关执行节点以及每个执行节点所需的信息，并将活动与执行它们的节点进行绑定。在整个系统中，每个节点都有一个节点管理器，它负责与过程定义节点通信，并管理过程表。图 4.2 给出了过程编译、绑定、过程实例与活动执行中的管理过程。

过程定义节点向每一个参与过程实例运行的节点管理器发一条消息，通知它与该节点有关的过程模型信息。当节点管理器收到过程定义节点发出的消息后，它将为该过程模型的每一个过程实例创建一张过程表，这张表里存储着过程实例运行所需要的静态信息。接着节点管理器创建一个线程负责维护这个过程模型所有实例的运行。这个线程的第一个任务就是生成一个该过程的消息队列，该消息队列管理本过程的所有实例，并负责接收有关该模型的所有实例运行过程中传递给该节点的消息。节点管理器还为过程模型的每一个运行实例都创建一张实例表，该表保存了过程实例运行中的数据，这张表是动态存储在内存中的，不占用硬盘存储空间。

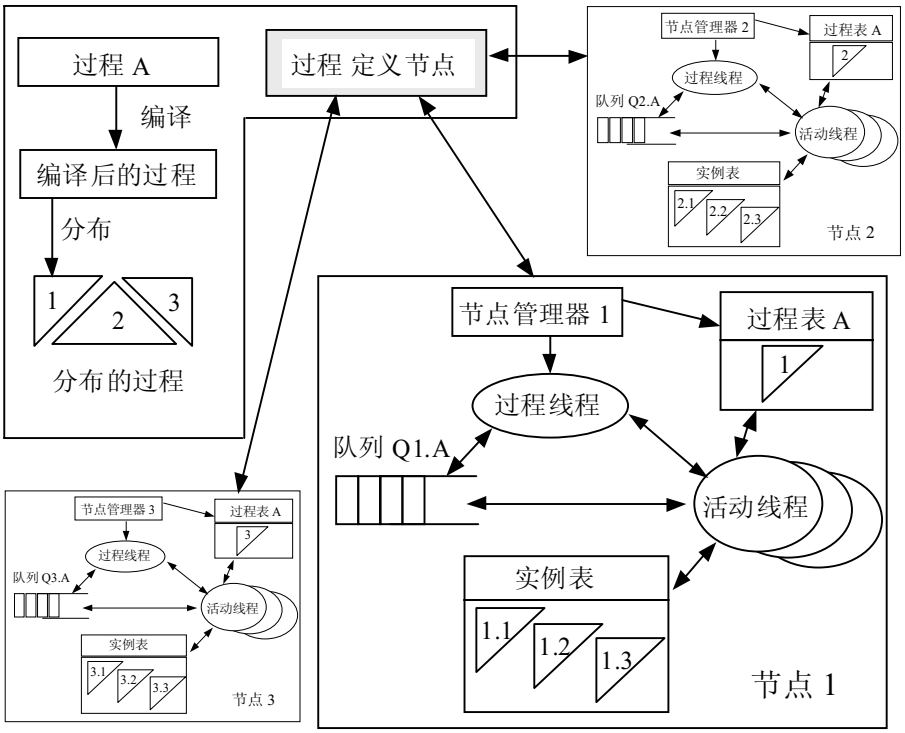


图 4.2 过程到不同节点的分布与执行

3. 过程实例的运行

节点管理器定时检查消息队列中的内容，一旦发现有新的消息，就创建一个活动线程，该活动线程负责对活动的运行进行管理。该线程运行的流程如图 4.3 所示。

该线程首先检查消息的内容（这时并不将消息从队列中取出，这种机制可以保证在发生异常情况时能够正确恢复），如果检查顺利通过，则确定需要执行的活动，然后从过程表中取出与该活动有关的静态信息，并在实例表中为该活动创建一条记录，描述该活动运行的状况，然后检查活动是否满足开始的条件。

如果该活动满足开始执行的条件，则激活相应的应用程序，并传递所需参数。当该应用程序结束后，活动线程对返回结果进行检查，并判断是否满足活动结束的条件，在满足了结束条件后则结束该活动，否则重新激活该应用程序。该线程还对后继的转移条件进行判断，并向相应的节点发送消息并传递输出参数。接着，实例表中的有关该活动实例的记录被删除，消息队列中有关该活动实例的消息也被取出。所有的这些操作都属于一个具有原子性的事务，这些事务特性通过系统对消息队列读、取操作的事务特性来实现。

如果有的条件目前还无法判断，活动线程会在实例表中记录已经判断过的条件，并进入睡眠状态直到有新的消息到达。

如果该活动开始条件不满足，系统会认定该活动应该被“终止”（非正常结束），并通

过其输出的转移连接弧向该活动的后继节点发出“dead-path eliminating”的消息，通知它们这个活动永远也不可能被执行；同时将该活动在实例表中的记录删除，并通过 GET 操作将该消息从消息队列中取出。所有终止该活动的操作也都属于与执行活动类似的原子级操作。与活动有关的所有消息将一直保存在消息队列中，直到整个活动结束。这样，一旦活动执行出现失败，还可以根据消息队列进行恢复。

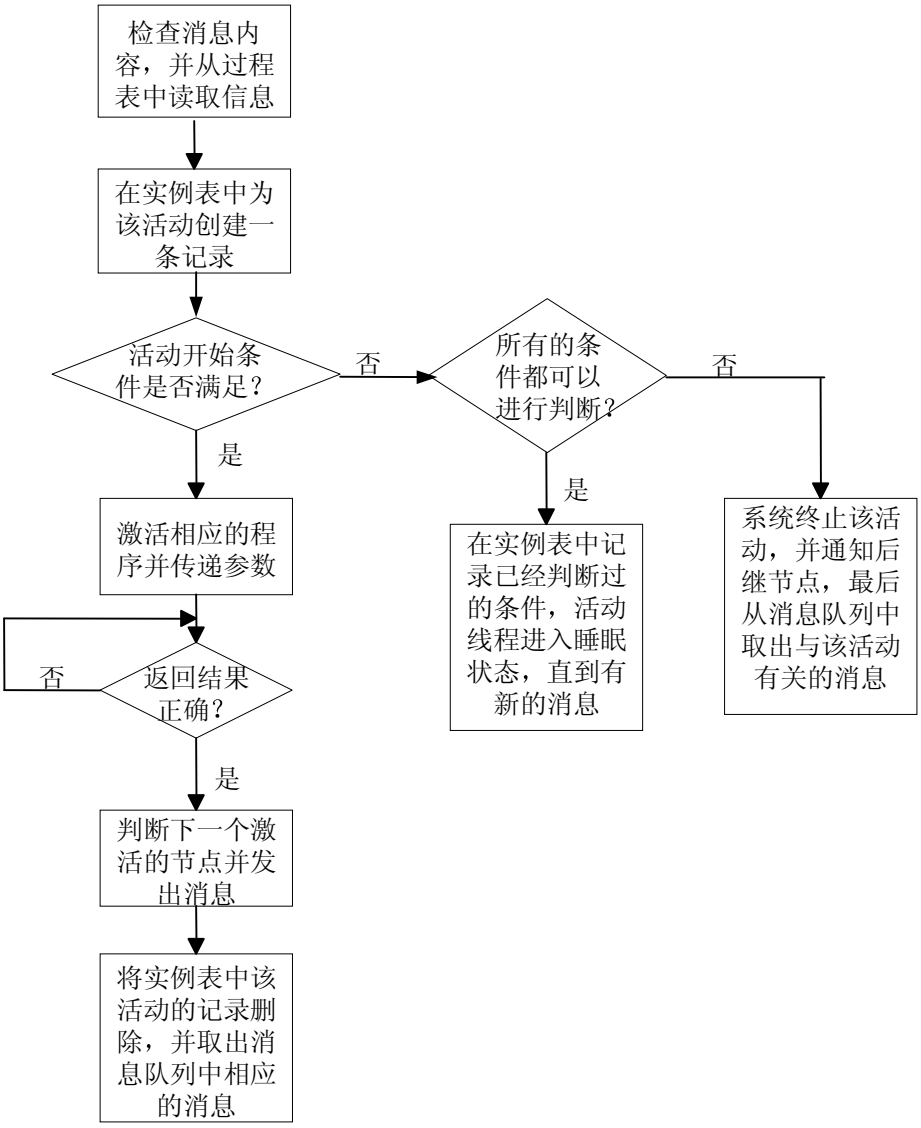


图 4.3 活动线程的运行流程

4.2.3 研究的关键技术问题

在 Exotica 中，主要研究内容集中在以下几个问题上[14]：

1．分布式工作流管理系统的鲁棒性问题

Exotica 研究了造成工作流管理系统失败的可能因素，并提出了一些解决问题的方法。

首先，在实例运行的过程中，参与活动运行的各方在活动运行前、活动运行中和活动结束后都要不断地传递消息。在 Exotica 项目中，项目组设计了可靠的消息传递机制来保证这些消息不会发生丢失，从而防止消息没有到达目的地或者活动运行的结果没有被记录下来。也就是说，所有的节点对活动的运行状态应该保持一致。在 Exotica 中，采用了多种方法来解决一致性的问题，如使用持久消息机制、为每个节点提供可靠的存储介质以及采用握手协议来保证所有的消息不被丢失。

第二种可能的失败情况是系统中某个节点在实例运行的过程中发生崩溃。如果是数据库发生崩溃，可以采用备份的方法进行恢复。如果是工作流管理系统中的某个节点崩溃，则需要将该节点应该完成的任务交给别的节点去继续完成。Exotica 采用了服务器簇（clusters of servers）来解决这个问题。每个簇都使用不同的数据库，每个数据库都保存了相同的信息，工作流实例可以运行在任何一个簇中。如果一个簇崩溃了，不会影响在另外的簇中开始一个新的实例。

2．工作流实例的恢复和取消问题

当工作流实例在运行过程中发生失败时，可以采用向前恢复（forward recovery）和向后恢复（backward recovery）两种方法来进行出错恢复。向前恢复指选择别的执行途径保证过程实例能够继续运行，向后恢复指消除失败的活动所造成的影响。Exotica 在这两个方向的恢复方面都做了研究，并允许在一个实例运行过程中多次改变出错恢复的方向。

3．备份问题

为了减小系统运行过程中系统失败对用户造成的影响，比较有效的方法是对过程实例的数据进行备份。如果对所有的过程实例都进行备份，大量的系统同步工作和数据交换会降低系统的效率。因此，Exotica/FMQM 提供了三种不同程度的备份方案：① 热备份－－将该过程所有变动都在操作的过程中进行备份，这种方法的优点是能够在发生出错的情况下，全面恢复系统，缺点是工作量大，系统中的每个状态更新都要在备份数据库中做相应的更改，系统负担重。 ② 冷备份－－只对过程运行中不同步骤的信息进行备份，当需要使用备份版本时，需要根据这些信息一步一步进行恢复，直至达到最新的状态。这样做会降低恢复的效率但却减小了备份的工作量；③ 正常情况下系统不进行备份，Exotica/FMQM 利用 FlowMark 提供的向前恢复机制，保证在服务器恢复后，过程实例能够从它过去中断处开始继续执行。在此基础上，项目组还对备份策略的实现方法进行了研究，提出备份机制，并给出了数据映射的实现方法和具体的备份算法。

4. 分布环境下的合作

目前有两种方法可以用来实现分布环境下应用的合作。第一种方法是在工作流系统的运行过程中将有关被执行过程的所有信息（包括模型定义和实例运行状态）都打成一个包，并在整个系统内传递这个包。这种方案的缺点主要是包含过程所有信息的包将非常庞大，加上为了实现活动的并行性需要在系统内将其复制多份，因此，它会对系统网络造成非常大的负担，降低系统的运行性能。第二种解决方案是对模型进行编译，事先确定执行的地点。Exotica 采用的是对模型进行编译的方法。这样，在运行过程中，需要传递的只是实例的状态和以前计算的结果，从而避免了由于大量数据传递造成数据库和服务器负担过重，成为制约系统性能提高的瓶颈。

5. 高级事务模型

工作流管理系统是一个支持异构分布应用环境的工具系统，这与高级事务模型的目标很相似。但工作流的语义更丰富、更适合应用于商业软件中。Exotica 研究了如何将工作流模型与事务模型相结合，为两种模型的进一步发展和应用都做出了贡献。

目前的事务模型，如 Saga、Flexible Transactions，对于用户来说过于复杂。Exotica 提供了一个中间件 Exotica/FMTM，它可以用作将复杂的高级事务模型转化成工作流过程模型的预处理器，其转换过程如图 4.4 所示。首先，用户创建一个包括所要使用的事务模型及所要执行的事务项的规范，预处理器在确认该规范符合事务模型后将它转化为 FDL（FlowMark Definition Language）格式的 FlowMark 过程模型。该过程模型包含了所有的活动和控制连接弧的信息。所生成的过程模型被输入到 FlowMark 中，在输入过程中，输入模块会检查过程模型中是否存在语法上的不一致，并生成该模型的 FlowMark 内部表示格式，所生成的内部表示格式在通过语义检查后，被翻译成一个可执行的 FlowMark 过程。该可执行过程将作为生成过程实例的模板使用。

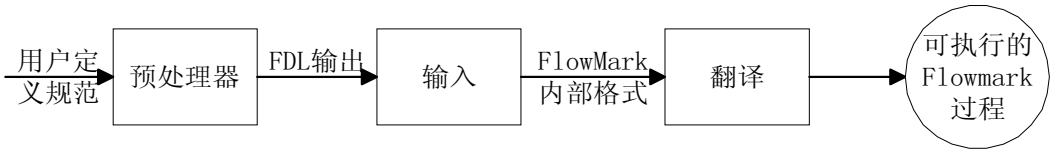


图 4.4 用户规范到 FlowMark 过程的转换过程

4.3 具有自适应能力的工作流管理系统-Meteor

Meteor（Managing End-To-End OpeRations）是由佐治亚大学计算机系的 LSDIS 实验室（Large Scale Distributed Information System Lab.）开发的一套工作流管理原型系统，其目的是实现一个能够支持大规模复杂应用的自适应工作流管理系统，并保证这些应用在异构企业环境中能够正常运行。该系统采用了完全分布的体系结构，在系统中设计并实现了分布调度模块来完成工作流任务的最优调度。它能够支持多种操作系统，如 Windows NT、SUN/Solaris 平台，并采用了 WEB 和 CORBA 等技术，为分布环境下的互操作提供了良好的支持。Meteor 对于各种异常情况的处理方法和系统出现故障后的恢复方法也进行了研究，提出了一系列措施来保证整个系统运行的安全性。在图形化建模工具建立的模型基础上，通过 Meteor 提供的代码生成器可以将工作流模型自动地转换为可供实际执行的代码，并且它还能够对模型进行动态修改，为企业经营过程的动态重组提供了有效的支持。

4.3.1 Meteor 的体系结构

Meteor 的系统体系结构如图 4.5 所示。整个系统由建模工具、自动代码生成器、运行环境（包括任务管理器和任务）、监控器和数据库（包括工作流模型库和运行数据库）组成。它为用户提供了两种运行环境（基于 CORBA 和基于 WEB）以满足不同用户的需求。

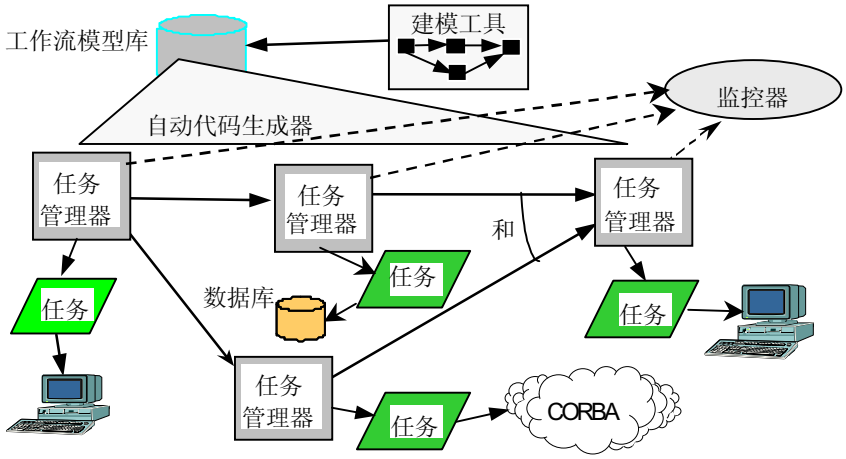


图 4.5 Meteor 体系结构图

在随后的几节中，我们先介绍 Meteor 的建模工具。在此基础上，对 Meteor 开发的两个系统 ORBWork 和 WebWork 的执行情况进行描述。最后对 Meteor 的研究方向和取得的研究成果进行介绍。

4.3.2 建模工具和工作流语言

Meteor 提供了图形化的建模工具，建模工具由三个部分组成：流程设计器、数据设计器和任务设计器。流程设计器描述了各个活动间的顺序关系，数据设计器描述了活动使用和传输的数据对象，任务设计器对每个活动的细节以及每个活动与外部应用开发工具之间的接口进行了描述。

1．流程设计器

流程设计器用来定义工作流模型中各个活动之间的关系。它采用图形的方式来表示工作流活动的流程。在 Meteor 提供的流程设计器用户界面上，节点代表活动，弧表示活动间的转换，一个活动的多个前驱活动之间可以使用 AND 或 OR 来进行连接。除了正常活动流程外，该流程设计器还提供了柔性的流程执行路径，即由用户定义一个在某个活动失败后流程可以继续执行的其它路径。例如，在建模期间，用户可以定义在一个活动发生失败时可以重试这个活动的次数，如果多次重试后依旧不能正确完成任务，用户可以定义一条路径，采取一些补救措施或其它方法来实现用户的业务目标。

2．数据设计器

数据设计器用于定义执行活动所使用和传递的数据。在 Meteor 中，采用了面向对象技术来实现数据设计器。建模人员使用数据设计器可以生成一个工作流流程中所需的所有数据类，并将其属性和对该类数据进行的操作进行封装。由于采用了面向对象建模技术，因此 Meteor 定义的数据类之间还可以进行继承，提高了数据管理的清晰程度，并强化了数据之间的逻辑上的关联。

3．任务设计器

Meteor 支持五种不同类型的活动：非事务型、事务型、WEB 型、人机交互型和两阶段提交型。Meteor 为不同的任务定义了不同类型的任务设计器。任务设计器可以有效的描述如何激活这五种不同类型的活动。对于需要从数据文件中获得输入数据的活动，任务设计器提供了一个模板来描述这些数据文件，模板中包含了有关数据类的属性和其它的一些文本信息。当工作流实例运行到该活动时，工作流执行系统会自动对这些模板进行初始化。

在模型建立完成后，模型将以工作流中间语言－－WIL （Workflow Intermediate Language）的形式保存起来。WIL 是 Meteor 自己开发的一套工作流语言，它与工作流管理联盟提出的工作流过程定义语言 WPDL（Workflow Process Definition Language）很类似。

WIL 能够记录活动间的前驱和后继关系、活动间所传递的数据对象、数据对象的定义、活动的详细描述以及活动激活方法等等。

4.3.3 工作流执行系统

在完成工作流模型建立后，还需要将模型转换成可供实际执行的代码才能提交给工作流执行系统进行执行。Meteor 为 基于 CORBA 的工作流执行系统和基于 WEB 的工作流执行系统分别提供了各自的代码生成器。该代码生成器以 WIL 格式的工作流模型作为输入数据，生成供任务管理器使用的可执行的代码，包括调度、活动激活、数据对象获取、以及恢复机制等代码。另外，对由数据设计器定义的数据对象，该代码生成器还可以生成对这些数据对象进行维护和操作的代码。

Meteor 提供了三种不同的工作流执行系统：ORBWork（基于 CORBA 规范的完全分布的工作流执行系统）、NeoWork（基于 CORBA 规范的集中式的执行系统）和 WebWork （基于 WEB 技术的完全分布的执行系统）。下面，我们对 ORBWork 和 WebWork 两种工作流执行系统进行分析。

1. ORBWork

ORBWork 是 Meteor 开发的基于 CORBA 规范的完全分布的工作流执行系统，它以支持 CORBA 规范的软件产品 Orbix 做为底层的通讯支持系统，使用 CORBA 规范中对象和 IDL 接口定义来实现系统中对象间的互操作和数据源的封装。在 ORBWork 中，系统的所有组成部分（包括任务管理器、任务、经过封装的应用程序、监控单元和恢复机制）都是以 CORBA 对象的方式实现的，而且一些原有应用系统也被封装成为 CORBA 对象，在执行时它们被当作任务（Task）调用。为了方便用户的使用，ORBWork 的用户界面都是以 WEB 页面的方式提供给用户，Meteor 还为此开发了基于 WEB 页面方式的客户端应用与底层支持系统 Orbix 之间的集成接口。

Meteor 的工作流执行系统主要包括任务管理器和任务两个部分。任务管理器是由系统的自动代码生成器根据在模型库中以 WIL（Workflow Intermediate Language）格式保存的工作流模型定义而自动生成的，它主要负责完成过程中任务间的导航以及激活相关联的任务，这其中包括应用数据的传递。每个任务管理器都是以 C++代码实现的 CORBA 对象，由于每个对象都负责所有工作流实例中相关任务的激活，因此，这些对象都是持久对象。

每一个任务管理器都是由执行前一个任务的任务管理器通过调用一个名为 Activate 的

方法来激活。Activate 是任务管理器对象所有 IDL 接口中唯一带有过程控制信息的方法，它带有足够的参数以便于后继的任务管理器能够进行它的工作，其中包括一个用于为所要激活的任务提供访问信息的数据对象名称列表。

任务管理器管理的功能主要包括任务的激活（task activation，指判断是否满足任务开始的条件）、任务的调用和监控（task invocation and observer，指调用相应的任务）、异常处理（error handling）以及错误恢复（recovery）四个部分。图 4.6 是一个简化的任务管理器。

当一个任务管理器接收到 Activate 调用时，它将执行以下动作：

1．保存调用者的 ID，并由此获得对所有数据对象的引用；

2．调用一个函数判断该任务执行的前条件是否满足，如果条件不满足则退出；

3．如果前条件满足，则查询所有需要用到的数据对象；

4．任务管理器调用 DoTask 函数来执行该任务，并等待返回结果，以判断是否出现异常情况。任务管理器还提供了超时处理机制。

调度信息存储于每个任务管理器中，代码生成器会根据模型定义中的的过程逻辑来生成任务管理器的调度逻辑。每个任务管理器都知道在某个任务结束后应该执行哪个任务。当任务正常结束后，该任务管理器会调用下一个任务管理器的 Active 函数。

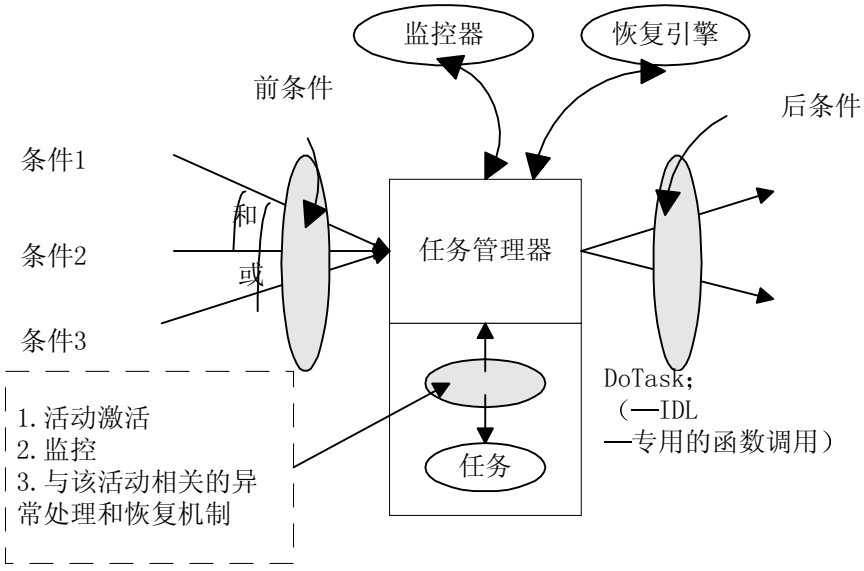


图 4.6 简化的任务管理器

在 ORBWork 中，任务管理器的设计采用了面向对象的技术。Meteor 设计了一个任务管理器基类 TaskManager，在此基础上，派生出基本任务管理器基类（BaseTaskManager），在此基础上又派生出了三类任务管理器：事务型任务管理器基类（BaseTransactionalTaskManager），非事务型任务管理器基类（BaseNonTransactionalTaskManager）和用户任务管理器基类（BaseUserTaskManager），它

们分别处理事务型、非事务型和用户活动。实际系统运行中的活动管理器都是从这三类中继续派生出来的，如图 4.7 所示。用户任务管理器（BaseUserTaskManager）允许用户添加特殊的处理程序。

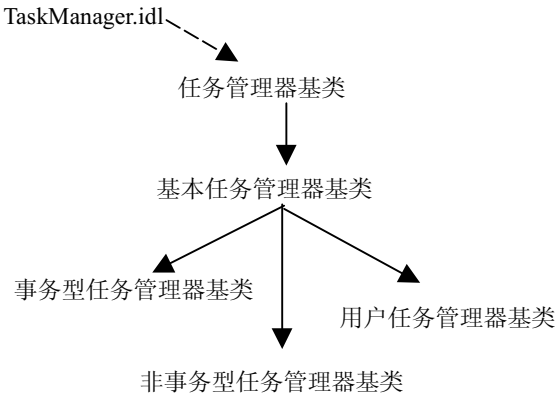


图 4.7 任务管理器的层次图

ORBWork 既支持需要用户参与的活动又支持自动执行的活动。ORBWork 为每一个需要用户参与的活动提供了一个 HTML 模板，该模板是在模型创建阶段由建模人员所定制的，其中包括该 HTML 页面中所需的数据对象的属性等信息。在工作流实例的运行过程中，当实例执行到该活动时，任务管理器将调出该 HTML 模板，获得模板中所定义的数据对象，并生成该模板的实例。这种机制使得建模期间的数据对象的定义和运行中实际的数据对象的使用相分离。WEB 和 ORBWork 之间的通讯通过 CGI 接口来实现，该接口负责将用户所有的输入信息进行收集并返回任务管理器。当用户提交一个页面时，系统会自动调用相应的 CGI 接口，该接口通过调用相应的任务管理器的 Done 函数来通知它任务已经完成（HTML 页面中的数据被封装在一个 CORBA 对象内）。

2. WebWork

Meteor 的研究人员考虑到企业可能出于价格、系统维护等原因不愿意去购买 CORBA 产品，但是大多数企业都有自己的 Web 服务器，或者可以连接到某个 Web 服务器上。因此 Meteor 开发出了一套基于 Web 的工作流管理系统。

与其它的工作流管理系统不同，WebWork 不仅仅是能够“支持 Web 技术”的工作流管理系统，而且还是“基于 Web 技术”的工作流管理系统。其差别在于，大多数工作流管理系统也支持 WEB 界面，但是他们采用别的通讯机制，如 RPC、CORBA、Socket 等等。一般来说，这些系统的工作流机都不是基于 WEB 的。这种情况下，工作流机是集中式的，

而客户端是分布的（单个服务器、多个用户）。我们称这种只提供 WEB 界面的机制为“支持 WEB 技术的”（web-enabled）。如果整套系统只使用 WEB 做为支持技术的话（也就是不仅提供 WEB 界面，还提供底层的通讯和系统分布支持），我们称这种工作流管理系统为“基于 WEB 技术的”（web-based）系统。

在 WebWork 中，代码生成器通常将模型中的一个活动模型生成三个部分：任务管理器（task manager）、应用任务（application task）和检验器（Verifier）。

任务管理器是将由模型定义所产生的 C++源代码进行编译得到 CGI 程序，其主要功能是从前驱活动中读取数据、为应用任务准备必须的数据、激活应用任务、从应用任务中收集数据并准备一个输出的 HTML 页面。它还负责对出现异常的应用活动进行处理和恢复。

应用任务负责完成各种具体的应用操作。它不负责工作流运行过程中的协作和控制，可以是由任何编程语言所编写的应用程序。

对于人工完成的任务，检验器由任务管理器输出的 HTML 页面内中置的 JavaScript 代码组成，它的主要任务是检查与用户的交互是否正确完成，并根据返回的结果确定下一步所执行的任务；对于自动执行的活动，它和任务管理器合为一体，负责对数据进行检查并确定下一步要执行的任务。图 4.8 反映了任务管理器、任务和检验器之间的关系。

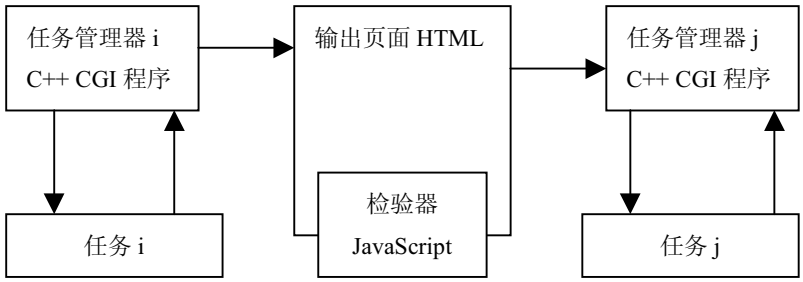


图 4.8 WebWork 运行组成成分

对于任务管理器的 CGI 程序来说，其处理过程如下：首先初始化数据（检查激活该任务的前条件），初始化函数读取前一个任务的数据，如果正常，则返回正确并执行任务。如果初始化不正确，则输出 FAIL。执行函数调用 DO_TASK 来实际执行该任务，除非在执行过程中出现异常情况（这时任务管理器将输出 FAIL），否则任务管理器将输出 DONE。整个流程可以由如下的程序段表示。

```
{
  if ( initialize() ) {                  //初始化
    if ( execute() ) {                   //如果初始化成功，则执行该任务
      output( DONE );                    //执行任务完成且无异常情况，输出“DONE”
```

```
            exit( 0 );                  //退出
        } else {                        //执行任务失败
            output (FAILED);            //输出“FAILED”
            exit(-1);                   //异常退出
        }
    } else {                            //初始化失败
        output (FAILED);                //输出“FAILED”
        exit(-2);                       //异常退出
    }
}
```

同 ORBWork 一样，对于不同类型的任务，WebWork 也提供了不同类型的任务管理器基类。实际的任务管理器都是从这些基类中派生出来的。

4.3.4 异常情况的处理和恢复机制

在 ORBWork 和 WebWork 中，都定义了几种不同类型的异常情况，对于各种异常情况都有专门的程序负责检测。当异常发生时，可以交给专门的异常处理函数处理，也可以执行一个补偿的活动来取消出错活动所产生的不利影响。

例如，在 ORBWork 中主要的异常情况有任务执行异常（任务执行过程中出的异常）、任务管理器异常（所有导致任务管理器异常工作的错误）以及工作流异常（活动间协作的错误，如通讯错误等）。异常情况的检测可以通过捕捉 CORBA 调用中的 IDL-exception 来实现，也可以通过轮流检测各个模块的运行状况来实现。

活动管理器和数据对象启动时需向本地恢复管理器（Local Recovery Manager，LRM）注册，当它们退出系统时再取消注册。而且每个 LRM 都有一个看门狗（watchdog）负责定时检查注册的活动管理器和数据对象是否还存在，如果发现有不能正常运转的节点，LRM 会重新激活它。重新被激活的节点能够自动与原来的节点保持状态一致，并向 LRM 注册。LRM 还会在本地日志中记录重要的事件，如活动管理器和数据对象的注册、取消注册等等。这些信息可以用来实现向前恢复。LRM 的失败由全局恢复管理器（Global Recovery Manager, GRM）来检测到。它采用和 LRM 相同的机制（看门狗和观察队列（watch list））来检测 LRM

的是否正常工作，并使失败的 LRM 能够恢复正常。

4.3.5 Meteor 与 Exotica 的比较

Meteor 和 Exotica 都是完全分布的工作流管理系统，每个节点都具有一定的自治能力，知道在某个活动完成后，应该执行哪个活动，这些信息都是在实例执行前进行编译（代码生成）时产生并存储在各个节点中的。它们的区别在于，Meteor 在活动结束后采用的是通过 ORB 进行函数调用来传递活动调用信息（调用下一个节点任务管理器的 Active 函数（在 ORBWork 中））。在 WebWork 中，检验器中包含了有关下一个活动的信息，由检验器（Web 浏览器）负责判断下一个应该执行的活动，并调用之。而 EXOTICA 则是通过消息传递来通知下一个节点执行。

除此之外，Meteor 和 Exotica 的编译机制也不相同。Meteor 有一个代码生成器（code generator），其功能是根据模型代码生成实际运行时的任务管理器程序。Exotica 没有这样的过程，只是在建模完成后，由建模节点根据模型定义中的节点信息将该节点执行活动所需的有关信息发给它，每个节点都一个节点管理器来维护，这个节点管理器是事先编制好的程序的，而 Meteor 则是动态生成任务管理器。所以 Meteor 在开发中更多的采用了编译器技术，其自动代码生成功能大大方便了其系统的应用实施。

4.4 基于分布式主动数据库技术的工作流管理系统-WIDE

WIDE（Workflow on Intelligent and Distributed database Environment）是来自西班牙、意大利和荷兰的五个合作伙伴联合开发的工作流管理系统。其主要目的是将分布式数据库和主动数据库技术应用于工作流管理系统中，为工作流的实现方法提供先进的、面向应用的技术支持。更详细地来说，WIDE 的主要目标有两个：

1) 为描述活动的流动和活动运行的环境提供一个先进的概念模型。它提供了功能强大的组织模型，并且对正常活动流动过程中出现的异常情况提出自己独到的见解，使得系统可以灵活地实现对不同类型异常情况的处理；
2) 通过采用主动数据库技术和事务管理技术为分布环境下的工作流管理提供了先进的技术支持。

我们首先介绍 WIDE 提出的工作流模型，包括组织模型、信息模型和过程模型。接着描述 WIDE 的体系结构，最后对 WIDE 利用主动规则技术对异常情况进行处理的方法进行

介绍。

4.4.1 模型的建立方法

WIDE 的工作流模型包括三个部分：组织模型、信息模型和过程模型。与工作流管理联盟提出的参考模型相比，WIDE 是对 WfMC 参考模型的一个扩展： WIDE 模型不仅定义了工作流的基本要素（三个模型及其它们之间的相互关系），而且还支持丰富的组织模型、复杂的活动分配约束、动态控制流程、复杂过程结构以及工作流事务管理。

1. 组织模型

组织模型记录了企业内组织结构和资源的信息。它不仅登记了单个雇员（人员）的信息，还包括人员所处职位的信息和为某个经营事务而临时组建的工作组信息，同时还记录了企业内资源的信息。组织模型还对雇员之间、职位之间和工作组之间的关系进行了定义。在 WIDE 的组织模型中，一个职员可以具有多个职位，可以参加不同功能的各种不同的工作组。在组织模型中定义了一组关系用来实现这些不同组织元素之间的关联。如包含（has-function）关系定义了组织功能与工作组之间的联系，分配（assigned-to）关系实现了雇员与职位相连，属于（part-of）关系将雇员与工作组连接起来。替代（deputy）关系允许雇员 A 代表雇员 B 去执行一个 A 本无权执行的活动。职位的层次图则描述了各个职位之间的上下级（职责）关系。在 WIDE 的组织模型中，还定义了代理的概念作为工作流活动的执行者。

与其它的工作流产品或研究项目所开发的原型系统不同的是 WIDE 严格实现组织模型与过程模型的分离，从而保证了这两个模型可以各自独立的进行发展。过程模型和组织模型之间的关系是通过授权机制来完成的，即通过授权机制实现过程模型中定义的角色到组织模型中代理的对应（Mapping）。

2. 信息模型

信息模型定义并维护一个工作流实例运行过程中所需的全部数据，并管理这些数据的使用范围和表示方式。从使用范围来分，数据可分为两种：全局数据和局部数据。全局数据指那些为所有的工作流实例所共享的数据，它们通常是以文件或数据库的方式进行存储。局部数据指那些只对一个工作流实例有效的数据，这些数据被该实例内的各个活动所共享。

WIDE 支持基本的数据类型，如整数型、字符串型等等。除此之外，它还提供了一个 article 的数据类型用于描述外部数据和对它们进行的操作。WIDE 还提供了一个 form 类型

的数据用于描述如何在客户端获取和显示信息。form 的最大特点是它包含了描述信息（包括标题和字段名称）和格式化信息。

3. 过程模型

过程模型中活动具有以下几个方面的属性：前条件（规定活动开始的标志）、活动所要进行的操作、后条件（表明活动结束的条件）、角色限制和对于系统定义和用户自己定义的异常情况的处理方法。

WIDE 支持层次化建模，在工作流模型中，活动可以是分层的，也就是上层一个活动可以由下层的一组活动组成。最底层的活动被称为基本活动，以区别于其它层次的活动。对于活动和基本活动的操作方式是不同的。对于活动来说，它所要做的是记录下一层活动之间传递的控制数据；基本活动描述了需要对信息模型中的数据进行的计算或用户执行的一些外部操作。因此，对于活动和基本活动来说，它们的角色限制属性也不同：活动的角色限制描述了下一层活动之间任务分配的从属关系，而基本活动则只限于本活动的内容。

WIDE 还定义了自己的一套规范语言来对三个模型进行描述。

4.4.2 系统的体系结构

WIDE 提出的工作流管理系统一共包括三层，如图 4.9 所示。

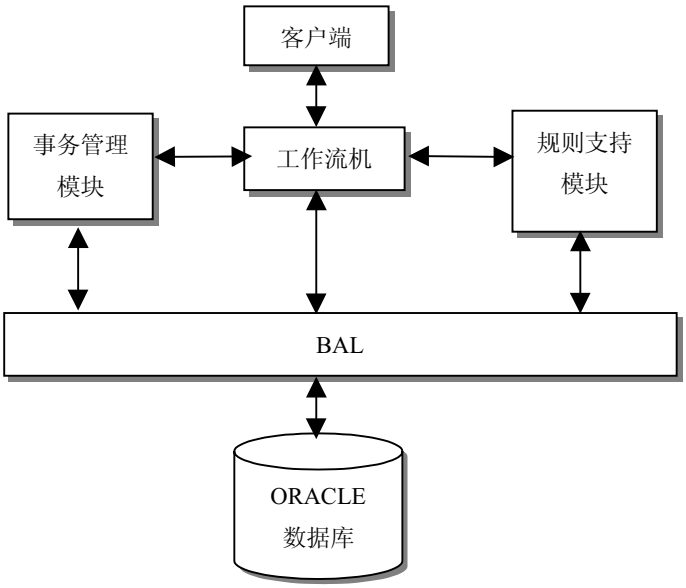


图 4.9 WIDE 体系结构图

最底层是数据库层，WIDE 采用 ORACLE 关系型数据库。数据库的屏蔽工作是由第二层基本访问层（Basic Access Layer，BAL）来实现。

BAL 为上层用户提供了一个面向对象数据库的操作接口，并将其映射为底层的关系型数据库操作接口。一个翻译器将面向对象的数据规则和定义转换成对关系型数据库的操作，从而实现它们之间的映射。BAL 层中包括负责对数据库查询进行处理的动态查询管理器（Dynamic Query Manager）、负责对象的插入、删除和修改等操作的对象映射器（Object Mapping）、为服务器提供数据库事件消息的变更通知器（Change Notificator）（主要是为了高层次的异常处理）和负责中间层次事务处理的本地事务管理器（Local Transaction Manager）。

服务器层位于最顶层，该层提供了事务管理服务和规则支持，同时还为系统用户提供了访问接口。整个服务器层包括工作流机、事务管理和规则支持三个模块。工作流机提供了规划自动化工具（Schema Automation Tool，负责任务间的导航）和本地调度器（Local Scheduler 负责角色的分配（如果本地的 Scheduler 不能满足要求，可以通过 Global Scheduler 来申请角色））。事务管理模块包括全局事务管理器和本地事务管理器。全局事务管理器（Global Transaction Manager）负责整个系统运行的事务处理，本地事务管理器（Local Transaction Manager）负责本地事务处理。规则支持模块主要指主动规则管理器（Active Rule Manager），它与时间事件发生器（Temporal Event Generator）（主要负责时间控制）一起负责对异常情况的处理。

4.4.3 异常处理策略与方法

WIDE 对于异常处理做了充分的研究，它使用主动规则技术（Active Rule）来描述异常情况及其处理方法。

1．异常情况的类型

在 WIDE 中，定义了三种不同类型的异常情况：

1） 警告（alarm）：在过程定义中已经预计到会发生的情况，但不是过程实例正常运行的结果。这种异常考虑到了基于应用的事件的发生，如一个特殊信息的到达或超时。

2） 工作流运行异常（workflow execution exceptions）：这种异常是活动的执行者没有按照模型定义来执行活动实例，如取消活动、取消整个过程实例、拒绝执行人工型活动、拒绝执行自动型活动、没有该角色等。在 WIDE 中，可以为不同的异常情况设立处理程序（exception handler），它可以接收参数，并按照定义进行处理。建模者可以自己预定义处理方法。WIDE 建立的系统为用户设立了一些缺省的异常处理程序，用户还可

以根据自己应用的需要添加异常处理程序。

3） 组织异常（organization exceptions）：组织人员变动对工作流实例的正常运转所造成的影响，包括对组织模型及其关系的改变、增加和删除。可以由建模者规定处理方法。

2. 异常的处理方法

在 WIDE 中，采用事件驱动的方法来对异常情况进行处理。最简单的方法就是采用 ECA（Event-Condition-Action）规则来实现，也就是当事件（Event）发生时，在条件（Condition）满足的情况下，执行某个动作（Action）。图 4.10 给出了简化了的 WIDE 主动规则支持模块的结构。

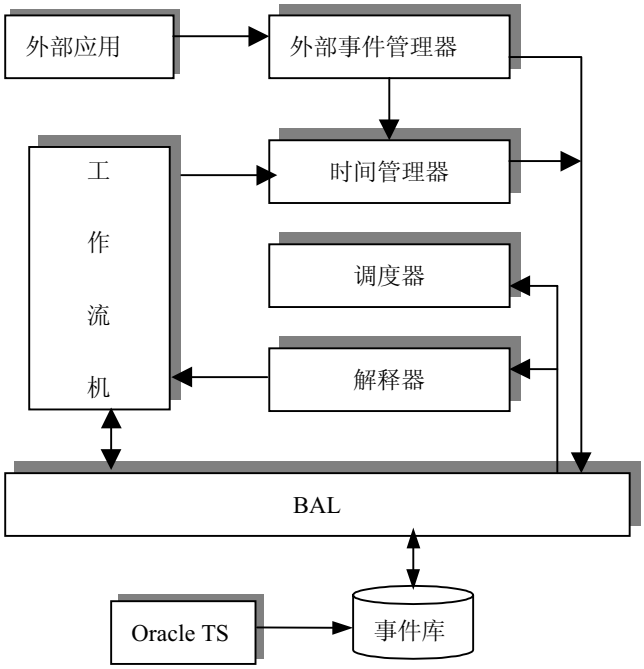


图 4.10 WIDE 主动规则支持模块结构图

在 WIDE 的概念模型中，共有 4 个事件类：数据事件（对工作流数据的改变，它也可以认为是工作流过程事件）、外部事件（由外部应用产生）、工作流事件（描述工作流的进展情况，如开始和结束）和时间事件（在工作流执行中直接或间接的时间点）。在 WIDE 中，数据事件由底层的数据库系统提供的触发器激活，外部事件由外部事件管理器激活，时间事件由时间管理器激活。外部事件管理器和时间管理器都将事件记录通过 BAL 写入事件数据库。

WIDE 将系统对于异常情况的发生、判断和处理方法写成一条条的规则，记录在数据

库中。当事件被激活时，系统根据数据库中记录的规则进行处理。规则的执行有两部分：规则的调度和规则执行。 调度器将数据库中关于事件的记录（通过 BAL）进行分析，判断哪些事件符合规则执行的条件，将其加入待执行的队列。解释器对规则执行的条件进行判断并执行规则。

4.5 基于状态与活动图的工作流管理系统-Mentor

Mentor（Middleware for Enterprise-wide Workflow Management）是由德国萨尔兰大学（University of Saarland）、苏黎士联合银行（The Union Bank of Zurich）和苏黎士工程研究院（ETH Zurich）合作的一项研究项目。其目的是为工作流模型的定义、执行和控制提供一个功能强大的中间件平台。

4.5.1 建模工具与建模方法

Mentor 采用状态与活动图做为模型建立的规范，并使用一个可视化的软件工具 Statemate 作为建模工具。执行活动中所需要的应用软件可以通过 Statemate 中的扩展接口（extension stub）集成到系统中。用户也可以使用 BPR 工具（如 Aris-Toolset）或其它工作流建模工具（如 FlowMark）来建立模型，Mentor 会自动将它们转化成状态与活动图。

活动图反映了系统功能的分解，每个活动代表了一个功能，这与工作流模型中的活动类似。活动之间通过有向弧进行连接，有向弧代表了活动之间的数据流动的方向和内容。状态图反映了活动之间控制信息的流动。在状态图中，状态的转换是由 ECA（Event-Condition-Action）规则驱动的。在状态图中还提供了嵌套的状态，即允许将一个状态图表示成一个高层的状态，在进入高一层的状态时也就自动进入了低一层状态图的初始阶段，当退出高一层的状态时，所有内嵌的低层次的状态也都自动退出。同时，状态图还提供了正交组件（orthogonal components），它允许两个处于相同层次的状态图可以互相独立地并行执行。

在实际的建模过程中，可以选择不同的建模工具。如果选用活动与状态图做为建模的规范，按照创建活动、将活动分配给角色、定义活动之间的数据流、定义活动之间的控制流这几个步骤进行，这个过程可以重复多次，直到整个模型建立完毕。同时，Mentor 还允许使用 BPR 建模工具或工作流工具来创建模型，Mentor 提供了转换工具来完成模型的转换。例如，用户可以使用 Aris-Toolset 来创建模型。接着，Aris-Toolset 可以将该模型转换

为 IBM FlowMark 的 FDL 格式。最后，用户可以将 FDL 格式的模型做为自动转换器的输入，将其转换成状态和活动图。

4.5.2 系统的体系结构

整个系统的体系结构如图 4.11 所示。

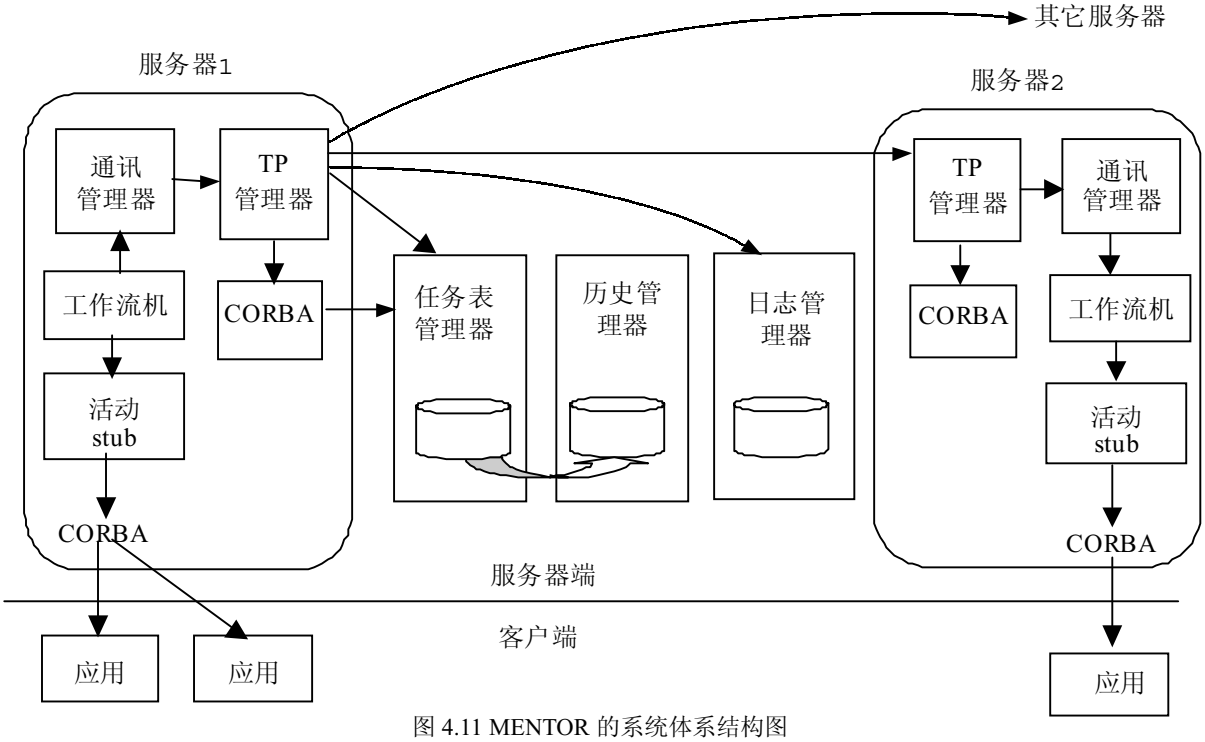


图 4.11 MENTOR 的系统体系结构图

为了支持异构环境下的各种应用程序，Mentor 开发的工作流管理系统以 CORBA 产品 Orbix 为底层支持系统，通过 ORB 来传递参数和激活应用程序。系统中所有需要被激活的应用程序都必须被包装成 ORB 对象，并提供 IDL 调用接口。在此基础上，Mentor 开发的工作流管理系统还包括以下几个部分：

1） 工作流机：负责活动间导航、查询执行活动所需参数、监控活动的运行情况等；

2） 通讯管理器：负责不同工作流机之间同步信息的收集和发送；

3） TP Monitor（Tuxedo）：为了支持容错，系统内所有服务器之间的通信都通过 TP Monitor 来实现。这主要是出于两个方面的考虑：首先，必须保证整个系统过程实例运行状态的一致性。这一点由 TP Monitor 提供的可靠的消息队列和在多个本地数据库和队列中读写操作的事务特性来保证；其次，需要有一个可靠的工具来管理整个运行环境中的各个部件。Mentor 采用如下的方法解决这一问题：工作流机和其它运行环境中的部件

在启动的时候都要在 TP Monitor 中注册，一旦这些部件发生崩溃，TP Monitor 会自动重新恢复该服务器；

4） 日志管理器：记录工作流中所有的状态改变、事件和控制流相关数据（control-flow-relevant data）的变化。它通过向 TP Monitor 发出事务型的请求，将向外发出的同步信息与日志管理器的记录绑定为一个事务，保证整个系统的一致性；

5） 历史管理器：用于记录目前正在运行和已经完毕的工作流实例的状态，主要是为了实现状态查询和日后的评估。同时，它也是任务项列表管理器的信息源；

6） 任务项列表管理器：负责将任务分配给能够完成此项任务的角色。它还能够完成其它的任务，如工作量的平衡、时间和优先权的管理等。它有一个本地的数据库记录相应的信息，而且它具有实时访问历史数据的权限。

4.5.3 工作流过程实例的执行方法

下面是 Mentor 在一个活动执行中和执行完毕后的处理过程：

1） 工作流机激活由它完成的过程实例中的活动实例（通过 ORB）；

2） ORB 将工作流机的参数转换为应用程序能够理解的参数，并激活应用程序；

3） 日志管理器记录本地工作流机的操作，这主要是用于恢复；

4） 当一个活动实例处于“ready”状态后，任务项列表管理器将该活动实例加入任务项列表中，并将该活动分配给一个可以完成此项任务的角色。角色的分配是根据任务项列表管理器的数据库中的数据决定的；

5） 历史管理器也将本地的数据库进行修改并进行记录。它进行这个工作的目的是用于查询、评估和对实例的控制。

下面几个步骤完成的任务是将本地工作流机进行的操作通知其它相关的工作流机：

1） 每当本地工作流机的系统配置发生变化后，工作流机都会通知通信管理器；

2） 通信管理器决定哪些工作流机应该接受变动通知，并提交给 TP 管理器。TP 管理器首先将同步信息和接受者加入持久性的消息队列，然后 TP 管理器将本地工作流机的更新和将同步信息加入消息队列捆绑成一个事务，保证系统更新和信息同步能够具有相同的结果；

3） TP 管理器将同步信息发送到各个工作流机；

4） 在各个工作流机的节点，TP 管理器将同步消息从本地消息队列中读出来（该消息也储

存在可靠的存储介质）。读消息和对本地进行修改属于同一个事务；

5） 本地的通信管理器将该同步消息传递给工作流机，然后对本地信息进行更新。

4.5.4 主要研究方向与关键技术问题

Mentor 的研究人员还对系统中许多其他的问题进行了研究，例如如何正交地分解过程模型使其能够在分布的环境下运行、系统的同步问题，并提出了相应的算法和实现方法。

1. 对工作流模型的正交分解

为了实现工作流在分布环境下运行，需要将工作流模型分解。这不仅是出于企业部门划分的考虑，也是实现轻负载工作流系统、充分利用企业资源的需求。

对一个工作流模型的分解需要进行两方面的工作：状态图的分解和活动图的分解。相对来说，活动图的分解容易一些。在 Mentor 中，活动图的分解是以地理位置（也就是企业部门分布）为分解的依据。由同一个部门所完成的活动将被分派到该部门的服务器上，在分解后的活动图中，这些活动都隶属于一个部门。状态图的分解则复杂一些。一个状态图可能是一个任意深度的状态树，而且，父状态所隶属的活动和子状态所隶属的活动可能不是由相同的部门所完成的。同时状态的转换也有可能将不同层次、不同活动的状态连接在一起。Mentor 在状态图的分解上进行了研究，提出了一套解决方案。他们还证明了该算法的正确性，也就是保证分解后的状态图在逻辑上和原来的状态图一致。有关这方面的详细的介绍请见参考文献[23，24]。

2. 系统的同步问题

在分布式系统中，整个系统的各个部分之间需要不停地进行同步和交换信息。在 Mentor 中，通过广播同步的信息和需要同步的数据的方式来实现信息的同步。同步信息的广播通过三个步骤完成，其中前两步是由发送信息的工作流机来完成的，最后一步是由接收信息的工作流机来完成的：

1）同步信息的收集：包括对正在执行的活动和状态图的更新和临时存储的更新信息。

2）同步信息的发送：将同步信息发送出去。

3）数据更新：信息的接收者使用同步信息将本地的数据进行更新。当所有的同步信息都收到后就可以执行下一步。

Mentor 的研究人员还就如何减少同步信息接收者和同步信息的大小进行了研究，并给出了一个算法，详细情况请参考[23]。

3. 系统的容错能力

对于系统的容错能力，Mentor 通过设置一个 TP Monitor 为系统提供可靠的消息队列，它还能够保证一个事务内部各项操作的原子性，并确认每条消息只被发送一次。同时，TP Monitor 还将记录系统中的各个部件的登录和退出情况，以保证当其中的一个节点失败时，TP Monitor 会自动重新启动它们。而且，在重新启动的时候，还可以参考日志中的记录将它恢复到上一次同步时的状态。

4.6 工作流管理技术的其它研究情况

除了上面介绍的四个主要项目外，多年来，在工作流管理技术研究的理论基础、实现技术与应用的三个方面还取得了许多研究成果。这些研究成果推动了工作流管理技术的研究、发展与应用。在工作流体系结构与模型定义上，国际工作流管理联盟做了相当多的工作，如提出了工作流参考模型体系结构、工作流管理系统的互操作接口定义、工作流定义语言等工作。在工作流建模问题上，国外的许多研究人员从各自的研究背景和应用需求出发，分别提出了许多有价值的方法。本节从工作流建模方法、工作流定义语言、工作流实现技术、与工作流管理技术相关的研究成果几个方面来简单介绍一下当今国际上一些研究情况。

1．工作流建模方法

在工作流建模上，主要的建模方法有以下几种：

1) 基于活动网络的建模方法：以活动和活动之间的关系为基础

2) 基于形式化表示的建模方法：如基于 Petri 网的建模方法。在扩展 Petri 网的基础上，Aalst 提出了工作流网的概念[26]，并对应用中的许多问题，如结构特性以及如何将工作流映射到 Petri 网等进行了研究。Aalst 还研究了使用 Petri 网支持企业经营过程重组的问题，提出了“What, How, and By Whom”的三阶段方法。“What”阶段研究分析公司和业务单元的主要功能，“How”阶段研究这些目的如何实现，如确定所需要的活动和活动的顺序，“By Whom”阶段分配这些活动需要的资源。

3) 基于对话模型的建模方法：Winograd 与 Flores 在语言行为理论的基础上提出了一种基于对话的工作流模型，这种工作流模型是从客户方与服务方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，Action Technologies 的工作流产品 ActionFlow 就采用了这种工作流模型。本书的第 5 章对工作流产品 ActionFlow 进行了介绍。本书的第 6 章中对基于对话的工作流模型有深入的介绍。

4) 基于状态与活动图的建模方法：如本章介绍的 Mentor 研究项目进行的工作。它介于 Petri 网模型和图形化模型之间。对于用户，它比 Petri 网容易学习和理解，在模型的正确性方面，它比图形化模型易于验证。其不利的方面是模型验证工作比 Petri 网模型麻烦，可视性效果（易理解程度）比图形化模型差。

5) 基于事务模型的建模方法：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。事务性问题在工作流管理系统中更为重要，因为工作流管理系统比数据库管理系统的操作要复杂得多，其活动的持续时间有时还很长，工作流执行过程中出现错误的可能性更大，而且这些错误造成的影响也比较大，因此从提高工作流管理系统的可靠性出发，建立基于事务的工作流模型具有十分重要的意义。在数据库事务模型的基础上，研究人员提出了多种高级事务模型（Advanced Transaction Model）用来描述工作流过程，包括嵌套事务模型、多层事务模型、Sagas、分支/汇合事务模型、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。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。ConTracts 模型[29]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；Amit Sheth 在对这些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[30]，他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。

6) 图形化建模方法：这是大多数工作流管理系统采用的工作流建模方法，其特点是用户界面友好，易于理解，并且在需要进行模型验证时可以将其转化为扩展 Petri 网模型来进行验证。

2．工作流描述语言

工作流描述语言首先是 WfMC 提出的工作流过程描述语言 WPDL 和 NIST 提出的过程

定义语言 PSL。这两个语言都为规范化工作流的描述方式作出了重要贡献，本书的第 7 章专门对这两个描述语言进行了介绍。除了它们之外，还要 WIDE 项目中提出的工作流描述语言－－WFDL（Workflow Description Language）， Meteor 项目中定义的工作流定义语言－－WFSL（Workflow Specification Language）及任务定义语言－－TSL（Task Specification Language），另外还有一种在 C 语言基础上扩展等到的 C&Co[27]语言。

C&Co 语言在 C 语言的基础上增加了描述工作流及事务处理的语法规范。其具体的实现方法是：

1) 任务被定义为返回 void 的函数，并且通过“trans”关键字将每个函数声明为一个事务。事务可以被提交和放弃。而且该语言还允许用户编写对一项事务的补偿动作，该动作在事务提交后，如果需要进行回滚时将被系统自动调用。
2) 通过引入“通信变量”来解决任务间的通信与同步问题。
3) 利用顺序语句、分支语句的格式来描述任务之间的顺序连接关系、基于数值的连接关系。
4) 利用类似于结构定义的形式来描述业务数据结构。

C&Co 语言支持过程的分层描述，可以很好的支持高级事务模型，它还提供补偿机制，提供并发任务的创建及通信功能。

3．工作流实现技术

工作流实现技术是工作流管理技术中研究得最多，也是成果最多的方面。在实现技术上主要考虑的问题有采用什么样的底层支持技术、如何提高工作流管理系统的可靠性（包括在出现错误以后如何可靠的恢复）、柔性（处理工作环境中发生的变化问题，如人员变化、过程结构变化）、以及大规模复杂应用的分布性问题。本节从基础技术角度简单介绍一些这方面的研究情况，有兴趣的读者可以从本书第 8 章和参考文献中进行详细的了解。

1) 基于 Lotus Notes 软件：许多工作流产品采用 IBM 的 Lotus Notes 作为其底层支持系统，采用 Lotus Notes 作为底层支持系统的好处主要是它提供了处理分布应用和复杂文档的能力，而且其可靠性较高。因为 Notes 本身就可以看成是支持群组工作的简单的工作流管理系统，所以它尤其适合于那些开发支持群组协同工作的工作流管理系统。本书第 5 章介绍的 Espresso 就是基于 Notes 开发的工作流管理系统。
2) 基于 CORBA：许多工作流管理系统采用 CORBA 作为底层支持技术，尤其是要实现分

布式工作流管理的软件，CORBA 更是首选的支持软件。如本章介绍的 Meteor、Mentor 项目就是基于 CORBA 作为底层支持系统来开发工作流管理系统的。

3) 基于 WEB：基于 WEB 技术实现工作流管理系统有非常明显的优势，它具有良好的灵活性，用户界面友好，所以近些年来，许多工作流产品不但是以 WEB 方式作为其客户端界面提供给用户，而且在开发完全基于 WEB 和 JAVA 技术的工作流管理系统。如本章介绍的 Meteor 项目的 WebWork。

4) 基于消息队列和 TP Monitor：消息队列和 TP Monitor 都为上层应用系统提供了可靠的消息通信和维护机制，为工作流管理系统的可靠执行和故障恢复提供了良好的保证，采用这种支持技术开发的工作流管理系统有 IBM 的 Exotica/FMQM、Mentor 等。

5) 基于主动数据库技术：基于主动数据库技术实现的工作流产品有本章介绍的 WIDE。

6) 基于 ECA（事件－条件－活动）规则：基于 ECA 规则实现的工作流管理系统以事件来驱动工作流实例的推进，事件驱动为分布式系统提供了一种统一的组件行为描述机制，它可以通过严格定义事件语义来保证工作流的正确执行以及对它的监控，另外以事件驱动为核心的工作流管理系统不仅可以处理复杂的工作流过程，而且还可以大大提高系统的柔性，这种柔性允许在工作流执行的过程中修改过程结构，提高了系统执行的灵活性。针对工作流过程执行中出现的变化问题，Kappel[28]等人提出了一种综合面向对象数据库技术、角色技术和 ECA 规则的工作流管理系统，该系统以面向对象数据库 GemStone 为基础，在 GemStone 的上层定义了基于 ECA 规则的主动部件和 18 个通用的对象基类，这些基类可以用来生成工作流模型。通过在对象中引入角色以处理工作流执行过程中人员的变化问题，基于 ECA 的主动部件用来柔性的处理工作流执行过程中的变化。基于 ECA 规则的工作流管理系统还有本书第 8 章介绍的 EVE 软件。

7) 基于可移动代理：基于可移动代理技术实现的工作流管理系统可以在一定程度上解决集中式工作流管理系统带来的性能瓶颈和单点失败问题，利用可移动代理的持久能力，可以间接保证系统的可靠性，而且这种计算模式特别适用于复杂的计算环境，如支持移动式计算。基于这种技术实现的工作流管理系统有本书第 8 章介绍的 DartFlow。

8) 基于扩展事务模型：事务工作流是由 Georgia 大学 Amit Sheth 等人在文献[30]中最先提出的，这一概念强调了与工作流密切相关的事务属性。一个事务工作流包含了多个任务的协作运行，这些任务可能要访问到异构的、自治的、分布的数据库系统。任务间的协调是通过基于相互依赖的控制流方法进行描述的，它为每一个任务定义了执行的先决条件，这些条件可以是基于其他任务的执行状态（比如某一任务是否提交、是否

退出或结束等)，也可以基于其他任务的输出参数，或者基于某些外部变量（比如时间)。Amit Sheth 提出了对事务工作流的一些有意义的处理方法，主要包括任务的定义、依赖与正确性规则、工作流的执行等。基于事务的工作流管理系统目前还处于发展的初级阶段，但是由于其在保证工作流管理系统的可靠性和出错恢复上有重要意义，因此，随着工作流管理技术研究的深入，事务工作流技术一定会得到充分的重视和发展。

第 5 章 工作流管理软件产品

目前，工作流技术引起许多企业的兴趣，工作流产品的市场每年以两位数字的速度迅猛增长。而且，随着信息技术和计算机技术的发展，工作流产品的供应商又及时地将新的技术融入工作流中，提高了产品性能，使得工作流技术不断完善。

本书第 2 章中已经给出了采用基于底层实现技术、所实现的业务过程、任务项传递机制三种分类方法对工作产品进行的分类。这里简单概述一下这些不同的分类情况。

根据所实现的业务过程，工作流管理系统可分为管理型工作流、设定型工作流、协作型工作流和生产型工作流。根据底层实现技术分类，工作流管理系统可分为以通讯为中心的、以文档为中心的和以过程为中心的三类。根据所采用的任务项传递机制的不同，工作流管理系统可分为基于文件方式的、基于消息方式的、基于 Web 方式的工作流管理系统、群件与套件系统。 这些不同的产品各有特色，而且许多新的产品还在不断出现，老的产品也在不断更新。

本章的后续几节介绍几个典型的工作流管理软件产品。它们分别是 IBM 公司的 MQSeries Workflow、Action 技术公司的 Metro、 FileNet 公司的 Visual WorkFlo、JetForm 公司的 InTempo 和 Pavone 公司的 Groupflow。

5.1 IBM 的 MQSeries Workflow

MQSeries Workflow 是 IBM 公司推出的最新工作流管理产品，是 IBM 的商业集成软件 MQSeries 中的一部分。它将经营流程从应用逻辑中分离出来，可以帮助企业用更少的时间、以更快的速度集成非常复杂的应用与资源，实现降低成本、减少错误、提高生产力，从而达到可以根据市场需求灵活地改变经营过程的目标。MQSeries Workflow 以 IBM 公司的消息服务产品 MQSeries 为基础，可以将分布在异构平台环境下的不同活动、系统和应用程序有机地集成起来，为在 Internet 环境下实现电子商务提供了良好的保障。

与其他的工作流产品不同的是，MQSeries Workflow 是以 IBM 的消息队列产品 MQSeries 为底层支持。MQSeries 为 MQSeries Workflow 中的各个部分提供可靠的消息队列。整个系统是通过消息队列来进行通讯联系，因此不需要 IDL 调用或 RPC 等其它通信机制。

值得指出的是，MQSeries Workflow 已经实现了与 BPR 工具 Holosofx 的集成，并且正在进行与 IDS、Metasoft 等软件的集成。与 BPR 工具的集成对于实际的企业应用具有重要

意义。企业可以采用工作流工具建模其业务过程，然后使用 BPR 工具分析并优化其业务流程，最后可以将优化的过程交给工作流执行服务来执行。

5.1.1 产品体系结构

MQSeries Workflow 的体系结构如图 5.1 所示。

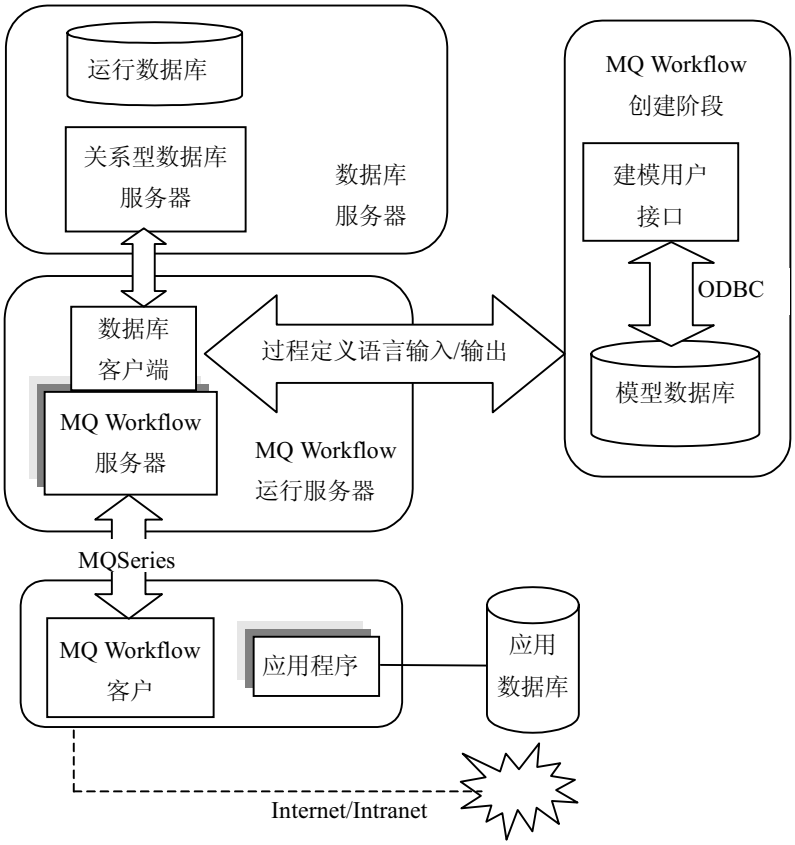


图 5.1 MQSeries Workflow 体系结构图

如图5.1所示，MQSeries Workflow采用了三层结构：数据库层（Data Server）、服务器和建模部分层（Server Components and Buildtime）、客户端层（Client Components）。用户也可以根据自己的工作机构的实际规模情况，将数据库和服务器层安装在一起，也就是采用两层的体系结构。下面分别介绍每一层的组成部分和主要实现的功能。

1. 服务器和建模部分

服务器协调和管理整个工作流系统和客户端的正常运行，它还负责记录并控制过程实例的运行状况。服务器包括执行服务器（execution server）、时间管理服务器（scheduling server）、清理服务器（cleanup server）和管理服务器（administration server）。执行服务器负责对过程模型进行解释，它创建过程实例并管理它的运行状况，它还完成活动间导航、

维护系统日志和过程运行状态，并维护任务项列表。时间管理服务器负责对有时间要求的活动进行管理，例如通知活动的执行者活动已经超时。清理服务器负责定期删除系统中已经执行完毕的过程实例的信息。管理服务器利用它的自恢复特性维护整个系统的一致性和负责异常情况下的恢复。MQSeries Workflow服务器充分考虑了扩展性，包括提供多处理器的体系结构，诸如SMP和SP2。多个执行服务器使用户可以将更多的工作流客户连接到同一个工作流运行数据库上，共享同一个工作流定义并运行过程实例。

MQSeries Workflow 的创建阶段（Buildtime）提供的建模工具允许管理员用图形化的方式描述一个过程和其中的活动，并且通过一个图形用户界面可以定义工作流模型中的所有其它信息。创建阶段提供的函数还可以定义 MQSeries Workflow 服务器的特性和它们的网络拓扑结构。图 5.2 是 MQSeries Workflow 创建过程模型的界面。在 MQSeries Workflow 建模界面中的图标可以由用户自行定义，这样可以使得用户方产生他熟悉的过程形象，如使用一个信封表示这个活动是发送邮件，使用一个机床表示一个加工活动，使用一个显微镜表示这个活动是一个检验活动等等。

作为建模工具提供的一个重要功能，MQSeries Workflow 不仅允许定义活动之间的控制流，而且允许定义数据映射。这个数据映射功能可以定义在过程实例的执行过程中，活动之间数据的传递和相关的传递规则中。

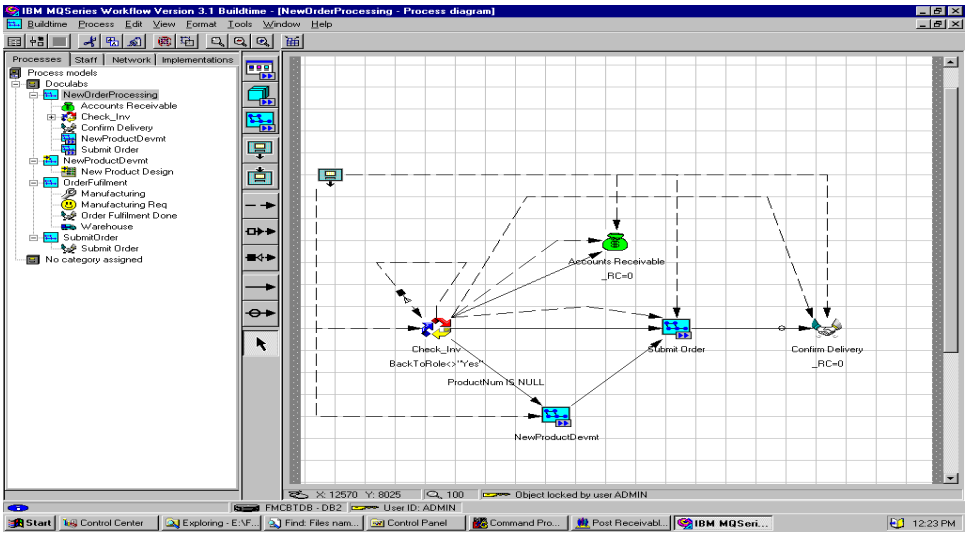


图5.2 MQSeries Workflow创建过程模型的界面

2. 客户端

客户端包括MQSeries Workflow客户（MQSeries Workflow Client）、管理功能模块（Administration Utility）和程序运行代理（Program Execution Agent）三个部分。用户通过

MQSeries Workflow客户来启动和监督过程实例的运行。管理功能模块为管理员访问管理服务器提供了一个用户界面，它允许管理员启动或终止服务器的每个部分，并查询它们的状态、查看不同系统的特性、并监控多个事件，它还允许管理员在MQSeries Workflow系统或系统组中观察它们相关的消息。程序运行代理负责激活用户在工作流模型中定义的应用程序并管理它们的运行。

图5.3是MQSeries Workflow的客户端界面，图 5.4是MQSeries Workflow的管理界面。

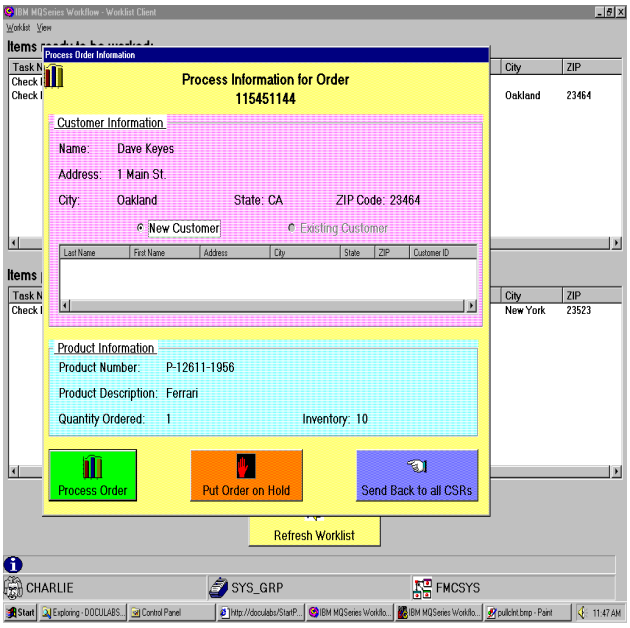


图5.3 MQSeries Workflow的客户端界面

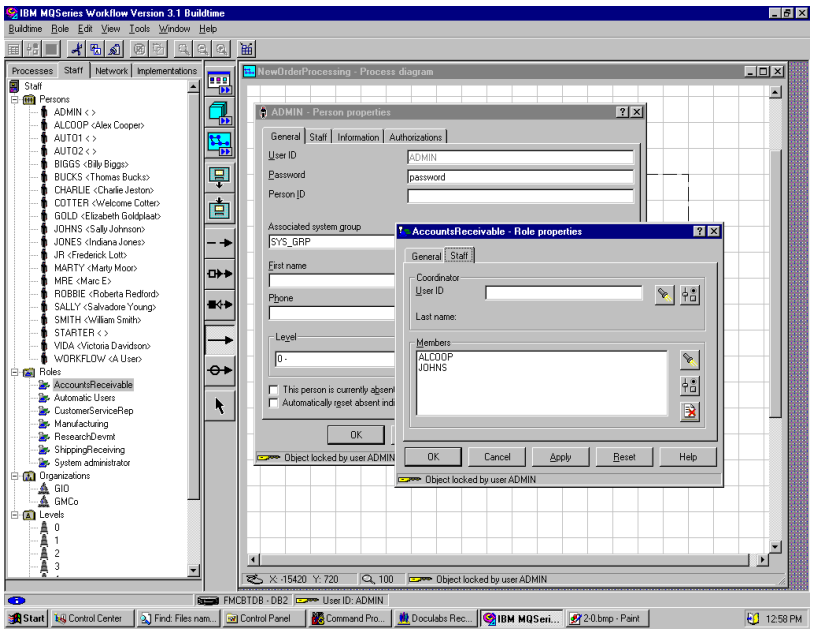


图5.4 MQSeries Workflow的管理界面

3. 数据库层

MQSeries Workflow选择了关系型数据库DB2存储过程模型信息和工作流相关数据，并为上层应用提供了相应的访问接口。

5.1.2 产品的主要特点

总体来说，MQSeries Workflow具有以下几个特点：

1. 方便现有应用集成与提升现有集成基础结构

IBM推出的消息产品MQSeries 可以支持25种不同的操作系统，这使得建立在其之上的MQSeries Workflow能够支持运行于不同操作系统的应用软件。用户可以将工作流过程扩展到企业甚至业务伙伴所使用的系统中，使电子商务（electronic business）的实现成为可能。另外，使用MQSeries 产品提供的集成器，MQSeries Workflow可以方便地实现与其它应用的集成。MQSeries 提供的预先配置好的应用模板可以大大缩短用户开发与集成应用的时间，加上MQSeries Workflow具有的面向对象的特性（包括过程、活动、触发条件、角色、人员的面向对象特性）使得这些部件具有非常好的可重用性。

在异构环境下，提升一个企业的现有集成基础结构对于它能够适应新的应用环境具有非常重要的意义。MQSeries Workflow 不仅允许用户保留其现有环境，而且可以利用它们来支持形成跨企业的工作流管理系统。

2. 支持复杂的组织与角色建模

MQSeries Workflow 允许柔性地进行雇员任务分配。用户可以根据组织结构或者工作流执行的历史等来选定，还可以允许定义雇员代替，即在一个工作人员不能完成其任务时，可以指定其它雇员来代替他完成任务操作。MQSeries Workflow 的推迟绑定（late binding）功能使得任务可以在工作流执行过程中将任务分配到人员。另外，MQSeries Workflow 还可以允许用户定义自己的客户端界面。

3. 自动分配任务

MQSeries Workflow可以根据模型定义自动地分配任务。当一个过程实例准备就绪时，活动被自动分配到指定的人员、部门或应用程序，并且有关完成活动所需使用的程序和相关的操作数据也会随着活动而传递给相应的人员或应用程序，从而达到缩短过程执行时间，提高业务过程执行效率的目标。

4. 图形化的界面

MQSeries Workflow还提供了图形化的编辑工具供用户来定义业务过程。用户可以使用简单直观的界面对过程进行建模。在MQSeries Workflow中，过程模型中包括了以下几个内容的定义：活动、活动之间的顺序、活动间的数据流动。MQSeries Workflow的建模工具提供了以下几个基本的模型元素：程序活动（表示执行一个应用程序）、过程活动（表示执行一个过程）、活动块（一组活动组成的一个模块，它表示重复执行这组活动，直到满足跳出循环的条件）、控制流和数据流。用户可以在建模期间将活动分配给特定的人员，也可以在工作流实例运行期间动态分配人员。MQSeries Workflow的组织模型中定义了三个不同的描述组织行为的方式，它们是组织（管理性的单位，如部门或项目组）、角色和人员。建模人员还可以把一个应用程序附在活动上，当活动的执行者选中该活动时，该应用程序会自动被激活。建模工具还提供了一些函数，供用户来定义工作流服务器的性质和网络拓扑结构。

5. 帮助用户执行和优化过程

MQSeries Workflow能够充分利用有关经营过程的知识，帮助用户定义、记录、测试、控制、执行和改进经营过程。用户可以迅速地创建、执行并优化经营过程，从而提高产品质量，提高效率，对市场变化做出积极的反映。

5.1.3 产品的应用范围

MQSeries Workflow 是 IBM 公司在 FlowMark 基础上进行改进后得到的工作流产品。它属于生产型工作流管理系统。FlowMark 的主要目标是实现企业的文档路由（document routing）和过程自动化（process automation），这也是传统的工作流产品的典型特点。MQSeries Workflow 不仅考虑到了企业对于数据传输的要求，还为在整个企业内人员、数据、应用和经营过程的管理提供了一个过程自动化的系统工具。它还允许外部人员通过 Internet 或 Extranet 参与到企业的经营过程中。MQSeries Workflow 能够支持多种操作系统，允许客户端和服务器实现跨平台操作，它能够对各种异常情况进行有效的处理。这使得 MQSeries Workflow 适合于需要大量人员合作和需要集成异构应用程序（如供应链管理、保险业、定单管理、信用卡管理等等）的企业和机构。MQSeries Workflow 的目标是方便地对企业内的各种经营过程进行重组，将企业内的各种异构的应用程序有机地集成起来，最终实现电子商务。

5.2 Action Technologies 公司的 Metro

Action 技术公司为知识工程师提供了一套基于 web 的工作流管理软件 Action Metro 4.0。它不仅能够管理确定的过程，并且对过程中不可预见的问题、要求和机遇也可以进行控制，还为用户提供了管理不确定的协作和任务的工具。用户可以用 Action 技术公司提供的工具在整个企业甚至企业间建立一个基于 web 或 C/S 的工作流管理环境。Action 技术公司的产品将 Internet 技术与 SQL 的事务处理技术结合起来，提供了内置的安全措施，即使对于最复杂的、广泛分布的和需要不断修改的工作流也能保证经营过程的一致性。其产品能够运行在多个服务器的分布环境下，支持各种工业标准，如 ODBC、Java、ActiveX、CORBA、 SMTP、MIME 等，具有良好的可扩展性和灵活性，从而可以保证用户在 IT 方面的投资得到充分的利用。

Action 技术公司的工作流产品基于对话行为的模型，在活动的请求者和活动的执行者之间要对要完成的活动达成协议，基于对话机制的模型强调让用户满意，而不是完成某一项任务。在对话行为模型中，在请求者和完成者之间需要四个交互步骤来完成一个活动：首先活动的请求者请求活动的完成者完成一项任务，接着双方对于要完成的活动和活动完成是否满意的条款达成一致，然后活动的执行者完成该项任务，最后活动的请求者对于活动的完成情况是否满意给出评价，活动的执行过程如图 5.5 所示。

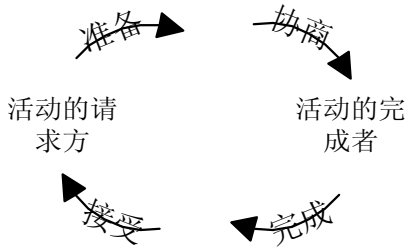


图 5.5 基于对话的建模方法

与传统的工作流技术相比，Action Metro 更适合于基于知识的工作流，它能够较好地支持用户与信息源之间的交互（如信息获取、用户协商、发现并解决问题、进行汇报等）。

5.2.1 Metro 的组成

Metro 整套系统由软件工具包、过程编辑器和服务器三个部分组成。

1. ActionWorks 软件工具包

ActionWorks 软件工具包（ActionWorks Software Developer's Kit）提供了一套原语性的 API，用户可以灵活地采用各种开发机制调用这些 API 来编制其特定的应用，如编制客户/服务器方式应用，或者编制遵循 DCOM 或 CORBA 规范的应用。

2. ActionWorks 过程编辑器

ActionWorks 过程编辑器（ActionWorks Process Builder）是一个用于设计工作流模型的图形化建模工具。用户只需进行简单的拖动操作就能完成整个模型的建立。它还为用户提供了快速应用开发服务，如界面的自动生成、工作流对象的定制、企业业务规则建立过程向导等等，使用户能够迅速方便地创建企业的经营过程模型。图 5.6 是 ActionWorks 过程编辑器的主界面。

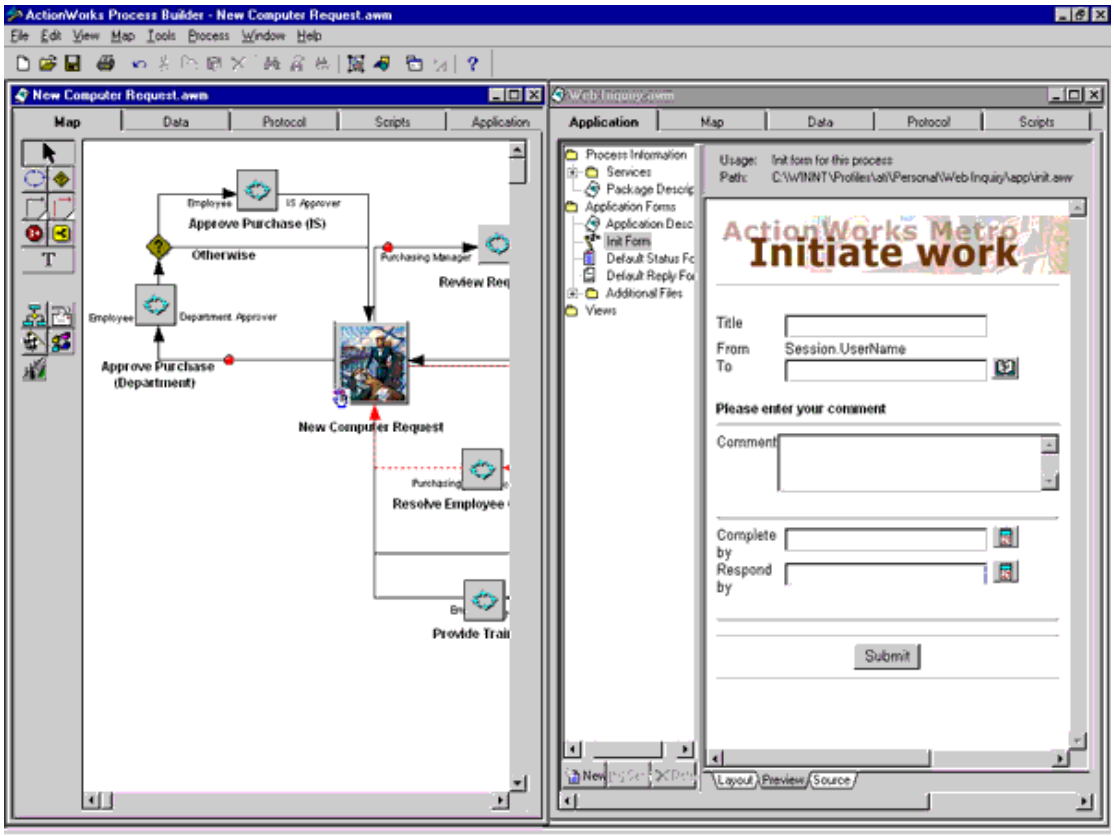


图 5.6 ActionWorks Metro 过程建模主界面

3. ActionWorks Metro 服务器

ActionWorks Metro 服务器（ActionWorks Metro Server）为活动的管理提供了一个应用服务器和基于 web 的活动管理工具 WorkBox 。图 5.7 是 Metro Workbox 的主界面。ActionWorks Metro Server 它能够管理各种不同类型的活动，如结构化的经营过程、基于团

队的合作项目和个人的不确定任务等等。它所提供的主要功能包括过程管理（如意外情况的处理，人员、角色和应用软件之间任务项传递等）、协作工作的管理（如支持不确定性的工作流，允许多个人员“共享”同一项工作，在最后期限到达之前提醒用户等）、监控功能（如获得状态信息、对实例状态进行强行修改等）。

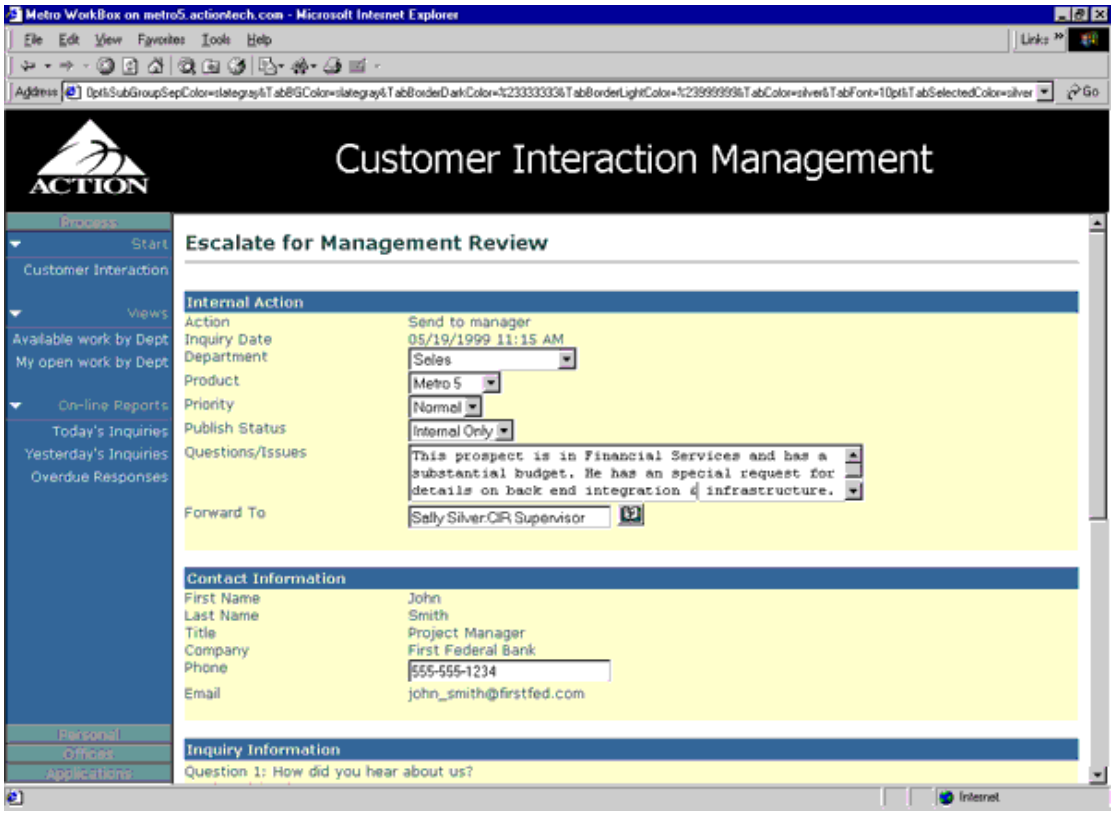


图 5.7 ActionWorks Metro WorkBox 主界面

5.2.3 Metro 的特点

Metro 主要具有以下几个特点：

1) 提供实时的状态信息：Metro 的开放式应用结构保证用户可以及时得到执行活动所需的最新活动状态、应用数据和应用工具，并且系统设计保证了不会出现由于数据冲突而影响过程一致性的问题。这样，用户能够根据实时的数据和过程状态作出决定。

2) 瞬间智能地分配任务：Metro 自动地将任务进行分类，用户可以通过基于 web 的工具箱（WorkBox）来接受任务。Metro 也可以通过 Metro WorkLinks 直接将任务送到用户的电子信箱中。在 Action 的解决方案中，用户不再是工作的被动接受者，他们是工作的管理者，并且，他们完全清楚整个经营过程，而不是仅仅了解他们所需要完成的部分。Action 的用户在接到任务时有许多选择。他们可以要求获得更多的信息、对活动

要求完成的期限时间进行商议、召集一个合作组，还可以将工作分给他人去做。

3) 提供过程一致性（Business Process Integrity）服务：当经营过程、用户和角色分布在多个服务器时，系统的复杂性也呈指数倍地增长。事务提交的冲突、数据被污染等问题将严重影响系统的正常运行。Metro 提供了过程一致性服务，过程实例可以在多个服务器上同时进行更新，并在必要的时候进行事务回滚。过程一致性服务主要包括过程回滚、过程一致性的加锁、异常情况管理、一致性检查、事务管理和关键点管理等。
4) 功能强大的过程编辑器：Metro 提供了一个图形化的过程编辑器，用户可以根据自己企业的实际情况很方便地定义任何类型的过程模型，包括可预知的结构化过程和需要协作的过程甚至不确定的工作流。过程编辑器还提供了代理执行器，所提供的代理执行器能够和 DLL、支持 ODBC 的数据库或者 OLE 进行通讯，并可以为用户自动执行任务。此外，工作流模板和协议向导可以帮助用户快速地生成过程模型。用户可以通过工作流模板定义简单的指令式的活动、复杂的协商过程、甚至对话类型的过程。协议向导则只需要用户回答几个问题，说明用户如何去完成任务，就可以帮助用户自动生成底层的应用、工作流实例和相应的数据库。
5) 过程评估和改进：Metro 能保证过程执行的每一步都有记录、跟踪和说明。它还能实时跟踪状态和收集数据。根据这些状态信息，用户能够迅速发现问题，并对过程模型进行改进和优化。
6) 支持 web 技术：Metro 允许用户通过 web 参与到工作流实例的运行中。

5.2.4 Metro 的应用范围

Metro 适合于各种类型的工作流应用环境，如管理型工作流。主要的应用范围包括：

1) 客户查询：网站查询管理、信笺管理等；
2) 重要的经营过程：如金融资产管理、抵押贷款批准、产品开发等；
3) 管理：如开支报告、购买、执照批准等；
4) IT 服务：项目管理、计算机供应等。

5.3 FileNet 公司的 Visual WorkFlo

Visual WorkFlo（以下简称VW）是FileNet公司集成文档管理软件系列产品中的一部分，它与集成文档管理软件中的其它产品合作，为工作量大、性能要求高的经营过程提供了一

个全新的可扩展的过程改进方案。它允许迅速地评价和改进机构工作方式。VW可以随时查看过程中关键的细节，实时掌握过程的运行情况。它还能帮助生成天、周、月、季度或者年的工作负载报告，以便改进过程模型。VW还提供 WEB服务，支持跨企业的经营过程集成，而传统的基于C/S机制的工作流产品则不具备这种功能。

VW 在实时查询过程或者活动的运行状态时，可提供事件日志和统计报告，预计瓶颈可能发生的地点，平衡工作量，优化执行过程，解决工作负担日益增长的问题。VW 还能够自动地确定每个工作流活动的参与者在某段时间内应完成的工作，提供有关工作量的信息，并能够根据活动的执行情况实现在相关人员之间的任务自动分配。

VW的解决方案是建立在基于组件的软件结构基础之上的，它将面向对象技术扩展到工作流应用开发中，提高了软件组件的重用率。在VW实施中采用了先进的工作队列处理方法，保证系统具有较高的可靠性。

VW主要由VW服务器、VW服务、VW设计模块、VW执行模块、VW监控模块组成。VW还针对不同的开发环境提供了面向不同任务执行程序的开发机制，包括：OLE接口，ActiveX控件和一系列API。用户可以自己选择采用哪种方式与VW相连。

5.3.1 Visual WorkFlo 的组成

Visual WorkFlo参考了工作流管理联盟提出的参考模型，整个系统包含以下几个部分：

1) Visual WorkFlo 服务（VWService）：对应于工作流联盟提出的参考模型中的工作流执行服务（Workflow Enactment Service）。它负责整个工作流实例的运转和维护。一个VWService可以有多个Visual WorkFlo服务器。
2) Visual WorkFlo 服务器（VWServer）：对应于参考模型中的工作流引擎（Workflow Engine）。VWServer是工作流实际执行的物理服务器。
3) Visual WorkFlo 组建器：对应于参考模型中的工作流建模工具（Process Definition Tool），用于建立企业的过程模型。
4) Visual WorkFlo管理器：对应于参考模型中的工作流管理工具（Administration & Monitoring Tools），主要负责对工作流实例进行系统监控和管理。
5) Visual WorkFlo执行器：运行于客户端的软件，能够支持两种客户应用：被动工作执行者和主动工作执行者，分别对应于参考模型中的直接调用应用（Invoked Application）和工作流客户应用（Workflow Client Application）。被动工作执行者是直接受Visual

WorkFlo 执行器控制的，不需要在应用中调用Visual WorkFlo 的应用编程接口。通常情况下被动工作执行者是那些不需要用户参与的应用程序。而主动工作执行者则需要在应用源程序中调用Visual WorkFlo 的应用编程接口，并且有可能需要使用基于ActiveX的控件，一般应用于那些需要人工参与的应用。

Visual WorkFlo针对不同的开发环境提供了不同的工作执行开发机制，包括：OLE接口，ActiveX控件和一系列API。用户可以根据自己开发环境的实际情况选择采用哪种方式与Visual WorkFlo相连。

5.3.2 Visual WorkFlo 的特点

Visual WorkFlo 具有以下几个特点：

- 状态跟踪和日志、报告的生成：Visual WorkFlo 可以实时查询过程或者活动的运行状态。它还能够提供事件日志和统计报告，预计瓶颈可能发生的地点，平衡工作量，为过程寻找最优，解决工作负担日益增长的问题。
- 软件重用率的提高：Visual WorkFlo的解决方案是建立在部件软件结构基础之上的，它将面向对象的技术扩展到了工作流应用开发中，提高了软件部件的重用率。工作流的设计者可以在原有的基础上对经营过程进行改进，而无需开始一个新的开发过程。
- 活动的自动分配：Visual WorkFlo可以自动地确定每个工作流活动的参与者在某段时间内应完成的工作，并提供有关工作量的信息。它还可以自动查询每个活动的状态，并激活下一个活动，将该活动分配给相关人员。而且，Visual WorkFlo还可以自动将用户与所需的Visual WorkFlo活动相连，将所需的应用软件和信息也随着活动传给活动的执行者，消除了寻找不同的程序和所需信息时容易发生的错误。
- 可靠性：Visual WorkFlo为企业工作流提供了较高的可靠性。它具有较强的鲁棒性，可以控制分布在整个企业中的用户和应用程序。它使网络传输减少到最小，并提供了较强的纠错能力。并且，Visual WorkFlo还提供了信息库，采用了先进的工作队列（Work Queue）处理方法，保证工作项不会被丢失，不会错过最后期限。
- 提供的web服务：Visual WorkFlo还提供了web服务，可以将经营过程扩展到传统的基于C/S机制的应用所不能达到的个人或其它企业。企业可以让使用工业标准的Internet web用户参与到工作流中，或者将目前的工作状态信息提供给客户和供应商。Visual WorkFlo的web服务是基于纯Java技术，并支持许多在不同平台上运行的应用软件。Java

的使用使得web服务器的选择具有很大的灵活性。用户的开发环境也可以各异，可以是纯Java，也可以在任何支持OLE的语言中使用ActiveX技术。

5.3.3 Visual WorkFlo 的应用范围

FileNet 公司的产品在过程类工作流产品中独占鳌头，它的市场份额一直位居榜首。应该说，Visual WorkFlo 还是比较适用于过程类的工作流应用环境。不过最近 FileNet 也开始将产品向文档类工作流类型靠近，它与 FileNet Panagon 的其它产品合作，构成了集成文档管理（Integrated Document Management，IDM）软件系列产品，为企业文档的集成化管理提供了一个新的解决方案。

5.4 JetForm 公司的 InTempo

JetForm 公司在企业工作流和电子表单解决方案（electronic forms solutions）方面处于领先地位。它的产品为企业在整个 Internet、Extranet 和局域网内实现经营过程的改进、管理费用的减少和产品竞争力的提高有显著的促进作用。InTempo 是 JetForm 公司推出的企业级工作流产品。它基于 C/S 方式，其客户端之间相互独立。它使用用户的电子邮件作为传递消息的工具，它能够自动将任务分配给相应的执行人员。InTempo 支持瘦客户应用环境，瘦客户应用将绝大部分应用程序的功能建立在服务器端段，从而可以大大减轻客户端的运行负荷。InTempo 还允许用户使用基于 web 的任务项管理器来参与到工作流中，这使得供应商和客户也可以通过 web 参与到企业的决策过程中。

5.4.1 InTempo 的组成

InTempo 整套系统包括：

- 组织定义器（Organization Builder）：用于建立和维护整个企业内员工的数据库；
- 角色定义器（Role Builder）：用于定义由谁来完成某项工作流活动；
- 过程定义工具（Process Builder）：负责定义工作流过程、活动和采取的行动；
- 代理（Agent）：负责通过电子邮件传递任务；
- 管理工具（Administrator）：用于控制代理的运行；
- 监控器（Monitor）：负责控制工作流事务的推进，并生成统计数据；

- web 连接器（Web Connectors）：用于将 web 浏览器与 InTempo 的代理相连。

5.4.2 InTempo 的特点

InTempo 主要具有以下几个特点：

- 客户端相互独立：InTempo 支持不同复杂程度的客户体系结构。它既支持简单的客户体系结构（如 Web 浏览器和网络计算机），也支持较复杂的功能强大的体系结构（如 Filler）。客户端的独立性使得企业可以为不同的任务选择合适的客户端环境。InTempo 允许在一个工作流实例中有不同形式的用户，如一部分用户使用 web 浏览器方式，其它的用户通过 Filler 的应用程序参与到过程实例的运行中。
- 支持多种数据库和消息管理系统：InTempo 支持很多的已有应用系统，如各种各样的电子邮件系统（微软 Exchange、各种支持 SMTP/POP3 的电子邮件系统等）、数据库（通过 ODBC 相连，如 SQL Server、Oracle 等）。这也使得异构环境下的信息系统可以很迅速方便地集成在 InTempo 提供的工作流环境中，这个能力也便于企业的业务和信息系统的扩展。
- 自动分配活动：InTempo 自动将任务分配给相应的人员，并使用电子邮件系统传输信息，任务会自动出现在用户的信箱中，简化了任务接收过程。
- 版本管理：InTempo 提供了对过程模型的不同版本进行管理的服务，使得用户可以在任何时间修改甚至删除掉一个工作流模型，而不会影响到正在运行的该模型的实例。版本管理保证正在运行的工作流实例可以按照旧的模型继续运行直至结束，而新的实例则将遵循新的原则。InTempo 会自动管理这一段过渡期，不需要管理员在版本控制上花费时间。
- 支持结构化和非结构化的数据：InTempo 不仅支持结构化的数据（如各种表格），还支持非结构化的数据（如用 Microsoft Word 编辑的文档）。用户可以将任何形式的文档附加在过程中传给下一个用户。
- 时间管理：InTempo 提供了不同程度的时间管理服务。用户可以在创建模型时加入期限限制。当期限已到时，InTempo 可以发出相应的消息。如果用户没有完成任务，InTempo 将会将该项工作从该用户的任务表中删除，并通知他的上司。也可以让 InTempo 在活动开始后的固定时间给用户发出提醒的消息。
- 支持不确定性的工作流：InTempo 允许在工作流中加入决策型的活动。对于异常和人

员参与的活动，InTempo 都将进行记录。这使得 InTempo 可以处理更为复杂的工作流，为企业更快、更准确地决策提供了保证。

5.4.3 InTempo 的应用范围

InTempo 适于管理型和不确定型的经营过程，它的应用范围主要集中在以下几个方面：

1．销售/市场：客户调查、客户投诉处理、贷款批准、产品改进建议的处理等等

2．管理：合同终止、资源调度等等

3．人员管理：内部调查、考勤卡管理等

4．金融/财会：财政预算批准、支出申请等

5.5 PAVONE 公司的 Espresso

自1994年初以来，PAVONE公司致力于开发和扩展基于Lotus Notes/Domino系统上的工作流项目以适应客户需求并在多种复杂环境下使用。这些项目运用PAVONE的Espresso工作流产品，充分体现了Notes在电子邮件(e-mai)和团队工作(team work)方面的显著优势。

Espresso工作流最初是在Notes版本3上建立客户/服务器的工作流产品。随着IBM公司不断扩展Notes客户(client)，特别是Domino服务器(server)的功能，使得工作流这一系统概念可以在Espresso工作流中得到充分实现。Notes/Domino完善的数据保密机制和分布式数据库之间的数据更新体制(replication)强化了PAVONE工作流产品的实用性和信息保密性。Notes/Domino在性能和可靠性上的持续增强更加促进了PAVONE工作流产品广泛应用。

Espresso集成软件是PAVONE的主要产品。它是由其早期的工作流产品GroupFlow和其办公室管理系统GroupOffice组合而成。Espresso拥有完整的工作流开发环境及日常办公室管理功能：如报告管理，地址管理和信件管理等。在此环境下，所有文档和信息经合理整理，每个工作人员都可以方便地访问与其有关的信息。Espresso的使用不受地域和网络限制。无论是通过当地(local)网络在家里，还是通过与内部网络(intranet)、外部网(extranet)或互联网(internet)的连接，用户在任何地方都能完成Espresso工作流的运行。由于Espresso提供了与Notes/Domino一致的工作流用户界面，用户可以通过Notes客户的内部网络，或运用web游览器的互联网，在其便利工作环境下使用Espresso， 如在饭店、家里、或者客户那里。

在Espresso提供的工作流开发环境下，用户可以快速简便地、以最少的编程费用建立

并推行具有复杂网络拓扑结构的工作流活动。在Espresso系统中不仅可以运行预定义的工作流，也可以定义和运行根据经营实际情况而临时设定的，由一组串行活动及其参与者构成的工作流。设定型(ad hoc)工作流不一定要在预定义的工作流基础上创建。

5.5.1 Espresso 的体系结构

PAVONE Espresso的体系结构如图5.8所示。

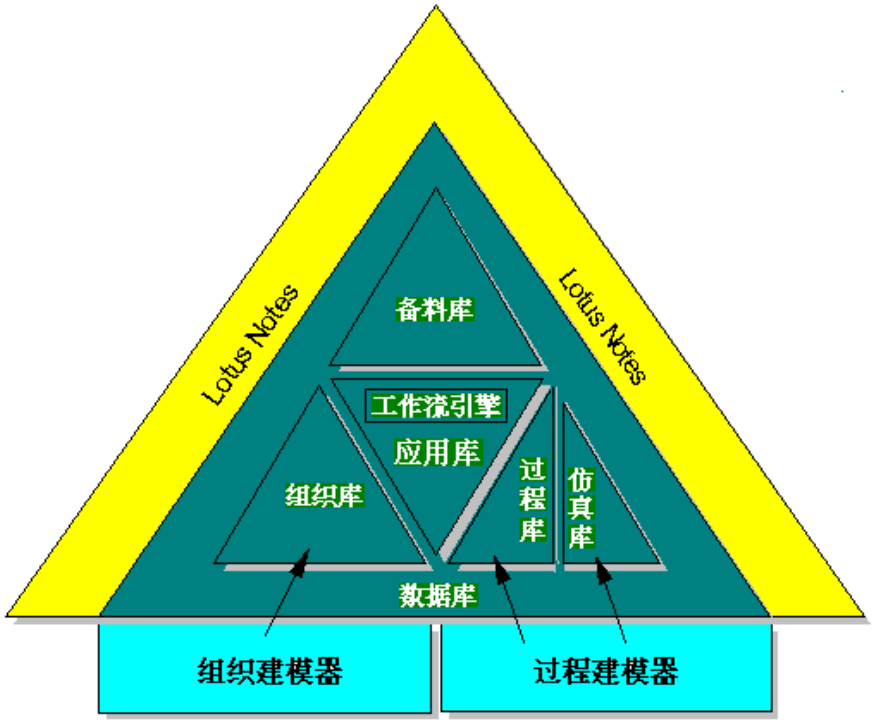


图5.8 PAVONE工作流产品Espresso的体系结构图

Espresso工作流系统是由两个图形建模器和若干Notes数据库组成。其中应用数据库是整个系统的中心。它作为用户界面是工作流的运行场所。PAVONE备料数据库(Settings Database)让系统开发者用来存放应用数据库中常用的工具和数据(如用于办公室管理系统的写信对象的各种称呼法)。

Lotus Notes/Domino是Eepresso系统执行期(run-time)的工作平台。Espresso的工作流引擎(PAVONE Process Engine)是装配在一个Notes/Domino的应用数据库(如PAVONE办公室管理系统数据库)之上的。它由LotusScript库(liabrary)以及Notes代理(agent)构成。工作流实例(instance)就在这个应用数据库上创建及运行。代表工作流及活动实例的Notes/Domino文档被分配到私人和公共的任务表中。如果一个工作流实例同时流向几个活动，就有几个活动实例文档对应于该工作流实例。这样并行活动实例就可以被同时执行。当有几个代表并行活动实例的文档流向一个活动时，那几个活动实例文档就会被合并成一

个。活动实例文档的合并既可以自动进行也可以由一个活动参与人员执行。用户可以在应用数据库上随时根据实际经营情况定义基于或不基于预定义工作流的设定型(ad hoc)工作流。在用户执行一个工作流活动时，他不需要在意该活动是预定义的还是临时设定的。

就象一般的Notes数据库一样，在包含PAVONE工作流引擎的应用数据库中也能调用外部应用软件(如MS Office和SAP等)以协助活动的执行。工作流活动实例文档中的信息能自动与外部关系式数据库(如DB2，Oracle等)进行交换。

Espresso工作流系统能在多种系统中运行，无论Lotus Notes/Domino是装在AS/400，OS 290，RS 6000，还是Windows NT平台上。只要有web浏览器，Espresso工作流可以在任何系统上都能使用。

1. 建模器及数据库

PAVONE过程建模器(ProcessModeler)用于图形化地建立工作流网络拓扑结构(network)和定义其他数据(见图5.9)。Espresso的工作流网络拓扑结构图是简单地由代表工作流活动的节点和连接两节点的单向流线组成。PAVONE过程建模器除了让用户按惯常手法随意画工作流结构图外，还可帮助用户利用 “自顶向下”(top-down)或 “自底向上” (Bottom-up)的方法结构化地创建工作流活动关系图。工作流活动不仅可以通过各种途径指派给人员，也能指派给Notes代理(Agent)来自动执行。

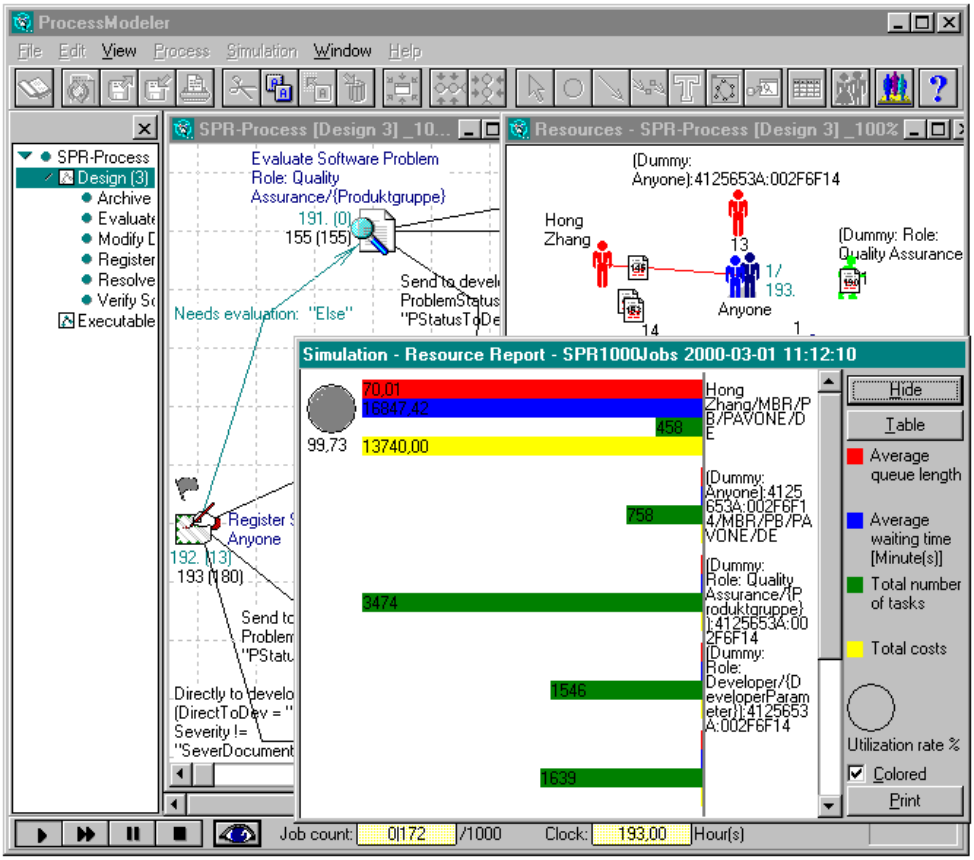


图5.9 PAVONE过程建模器及其仿真过程和仿真资源报告图

工作流定义的合理合法性可以通过采用过程建模器的检验仿真功能来探测和分析研究。为仿真而设置的有关工作流和资源的数据(如各种分布，资源参与某工作流的时间等)存放在一个仿真数据库中。仿真方案及其结果也是存入此数据库。结构固定的预定义工作流存放于PAVONE过程数据库(Process Database)。它在执行期间被应用数据库中的PAVONE工作流引擎读取。

Espresso中的另一个图形建模器PAVONE组织建模器(OrganizationModeler)用于方便直观地建立企业的组织机构数据并存入于PAVONE组织数据库(Organization Database)(见图5.10)。PAVONE组织数据库中包括人员(person)，部门(department)，工作组(workgroup)，角色(role)和物资(material)。它们及其候补可以被指派给工作流活动。人员和物资是执行工作流活动的基本资源。部门是一个企业人员的层次结构。工作组是为完成特别项目而构成的一组人员。角色是一组具有特殊技能或执行专项工作的人员。它可以带有一个参数以便于在工作流执行期动态地确定某个活动的参与者。

PAVONE组织建模器能读入Notes/Domino的组织目录(Domino Directory或Notes Address Book)以及LSA等组织数据库的数据并把它们转化后存放在PAVONE组织数据库中。PAVONE组织数据库和Notes/Domino的组织目录能直接被PAVONE的过程建模器和应用数据库中的工作流引擎利用。

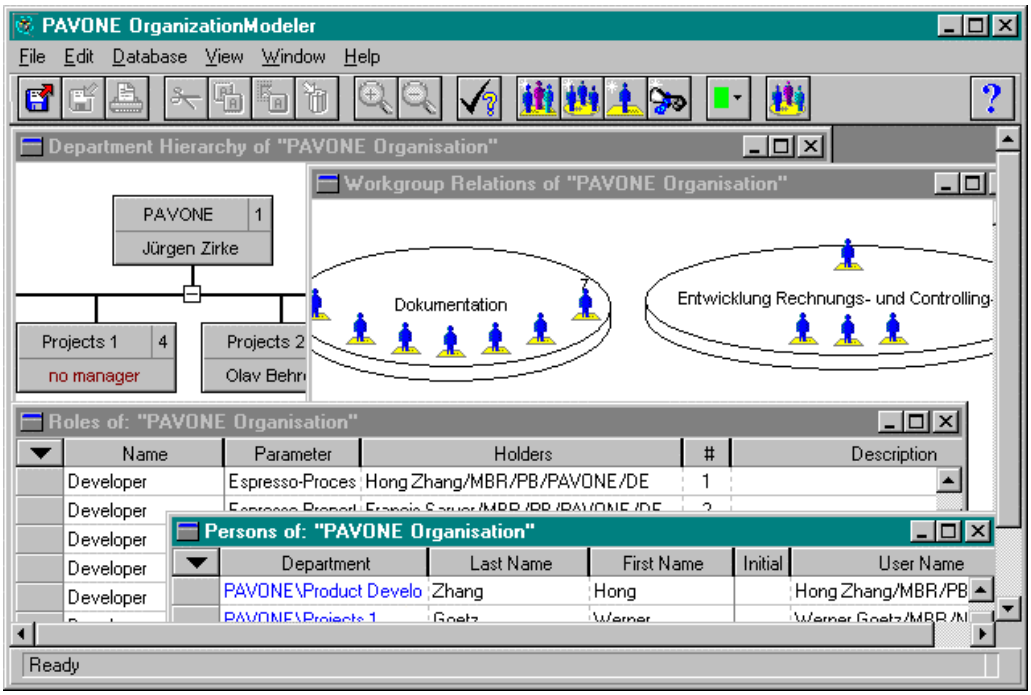


图5.10 PAVONE组织建模器：部门层次图+工作组图+角色表+人员表

2. 客户端

Espresso工作流的用户可以在其熟悉方便的Notes/Domino用户界面(见图 5.11) 上执行任务。它既可以执行复杂的预定义的工作流活动，又可以执行临时设定(ad hoc)的简单的工作流活动。这些活动不是强制性的。用户可以在任何时候创建预定义或者设定型工作流实例和执行工作流活动。当他创建工作流实例时，他只能见到所允许创建的预定义的工作流模型。预定义工作流实例的创建者必须属于工作流起始活动的参与者。为防止发生数据冲突，保持数据的一致性，用户在编辑一个文档以前可以对该文档加锁 (reserve)，不让他人同时编辑此文档。

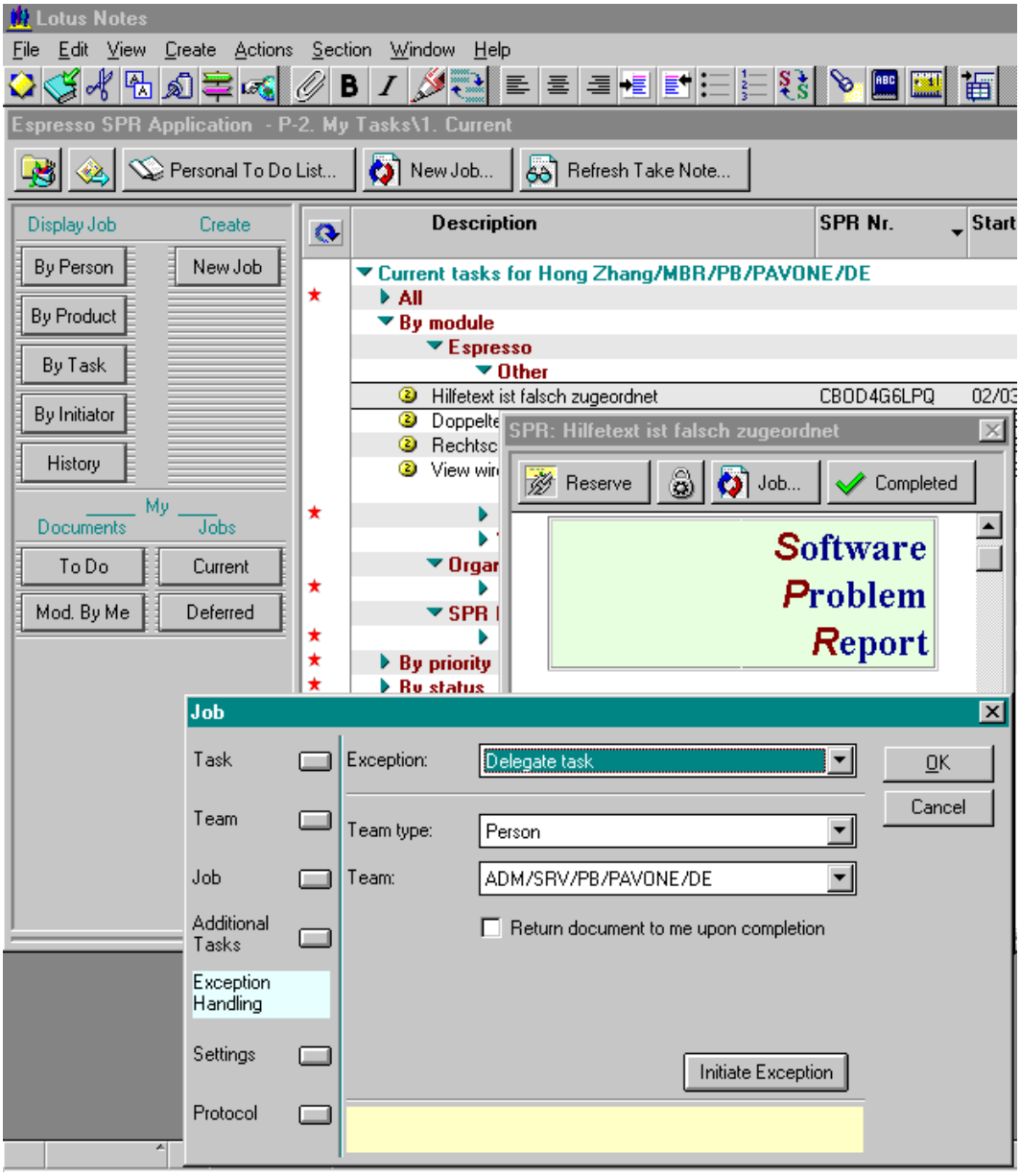


图5.11 Espresso 工作流系统用户界面：任务表/工作流活动/工作流对话框

Espresso应用数据库中提供的个人任务表是一组排列在一起的代表工作流实例的文档。它让工作流参与者只看到允许他执行的任务。工作流参与者可以象在其他Notes/Domino视图(view)下一样选择其中的任何一个任务执行。个人任务表可以按各种类别排列以满足个人爱好及工作需要。应用数据库中Notes/Domino表单(form)用于显示代表工作流活动实例的文档。工作人员也可以通过web浏览器直接访问Espresso应用数据库和执行工作流活动(见图5.12)。

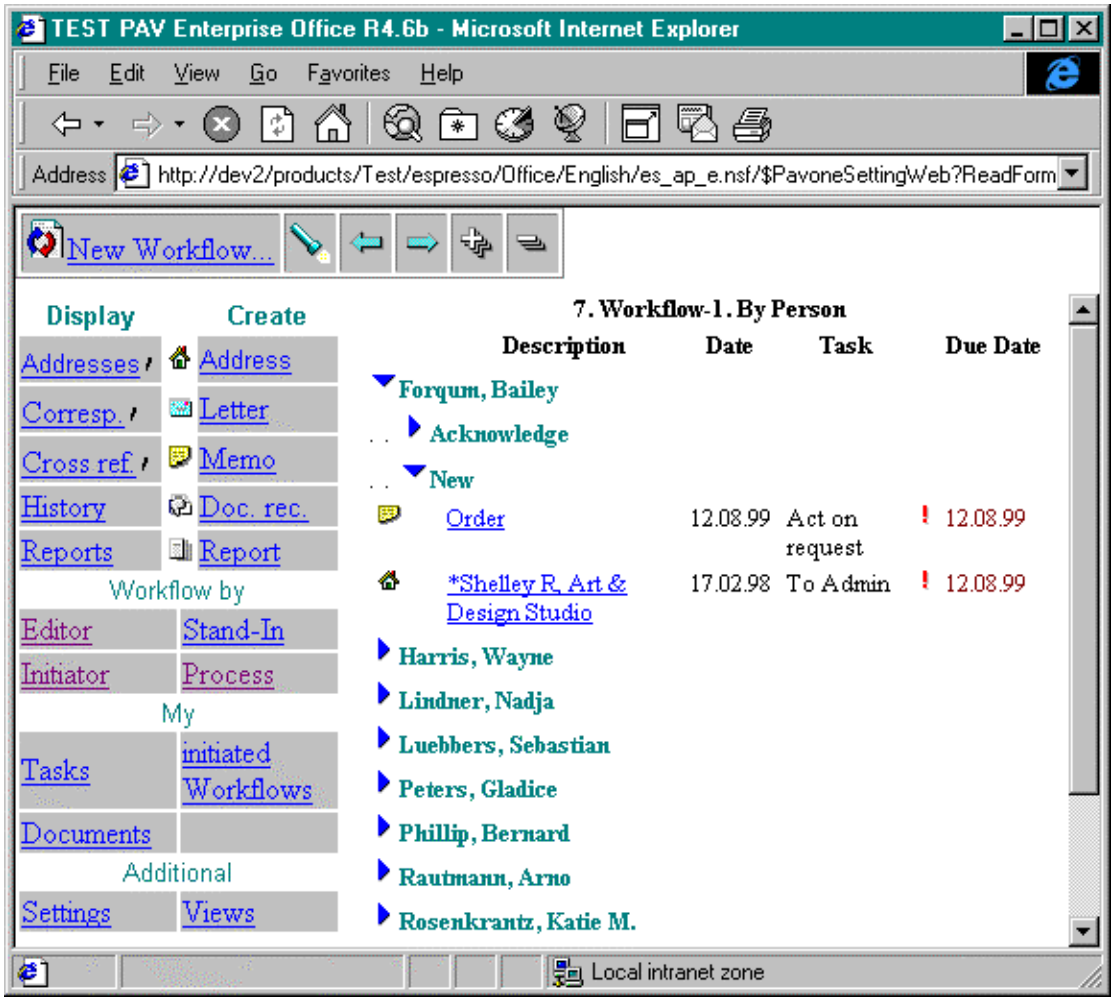


图5.12 Espresso工作流系统Web用户界面: 工作流任务表

为帮助一些重要的工作流活动及时被执行，相应的电子邮件(e-mail)将自动地发送到参与者在Notes/Domino平台上或在其他设置的邮址中，以通知他在任务表中有新任务。他可利用提供的文档链接(document link)或URL链接(URL-link)直接打开该任务的文档。

系统分析员利用PAVONE过程建模器可对一个预定义工作流在某个应用数据库中的实际运行情况随时进行观察和各种图表分析以便对存在问题的工作流实例及时进行管理控制。

过程建模器能读入运行中工作流实例的重要数据(如创建者，持续时间，当前活动，可执行者等)， 用户可以选择所关心的工作流实例进行统计分析，系统会将统计分析结果以直观的形式显示出来(见图5.13)。

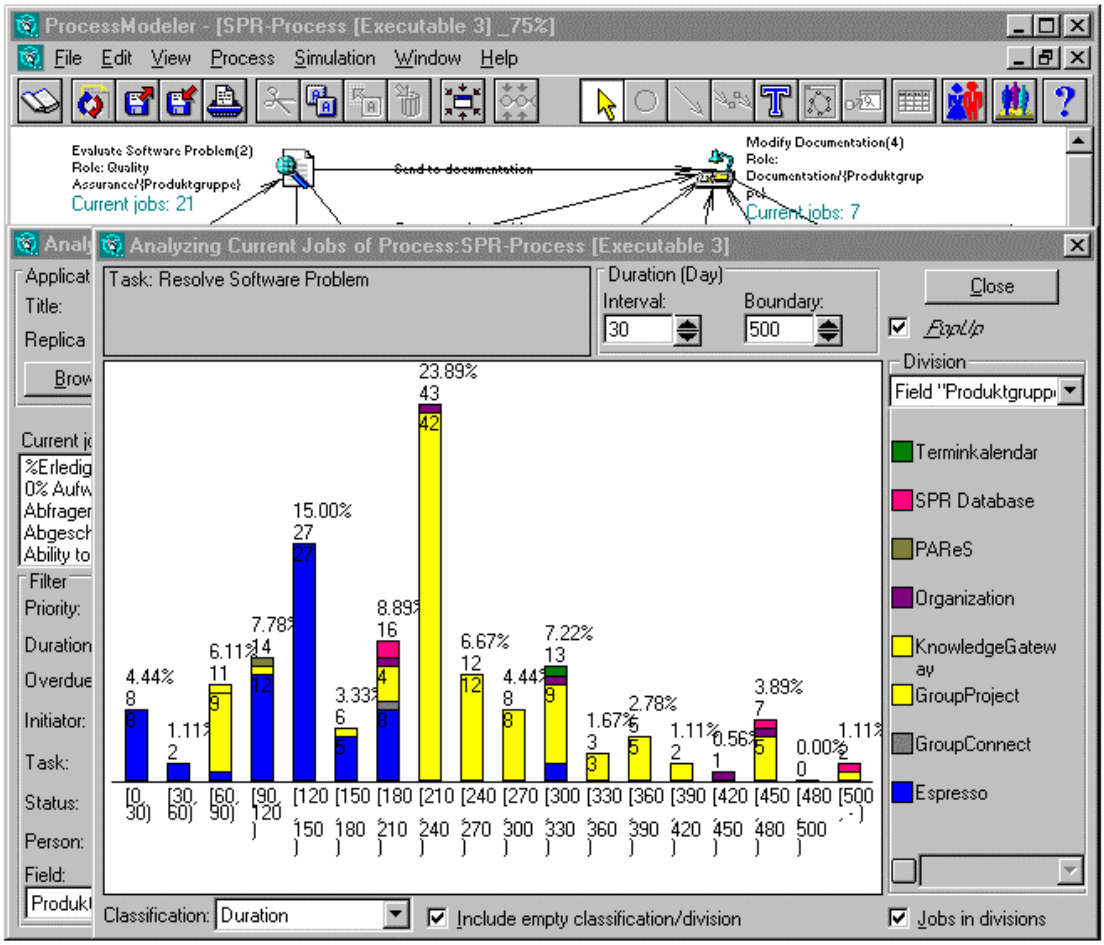


图5.13 PAVONE过程建模器上工作流运行分析界面

5.5.2 Espresso 工作流的特点

1. 简便快速的开发过程

简便的图形化的过程和组织建模工具，加上日常业务中常用的办公室管理系统应用数据库及其备料库使得Espresso工作流系统可以在一两天内在一个企业中推广运用。利用PAVONE的过程建模器，用户可以轻松地建立起工作流模型，并马上在应用数据库中得到使用。

由于Espresso工作流运行系统是在Notes/Domino平台上开发运行的，用户可以很方便地利用LotusScript和JavaScript等构筑用于专项业务的带有PAVONE工作流引擎的应用数据库。如果用户已有现存的用于专项业务的Notes数据库，在此数据库上装配上PAVONE工作流

引擎则能立即实现业务流程的自动化。

2. 工作流运行的机动性

PAVONE工作流引擎中的临时设定功能(ad hoc)和例外处理功能(如变更活动执行者；取消工作流实例；在同一个或另一个预定义工作流中选择其它活动等等)使得一个工作流实例不一定要按照或严格地按照预定义工作流运行。因此用户可以在Espresso系统中机动处理例外事件。

3. 预定义工作流的通用性

若Espresso预定义工作流中的活动都不是指配给特定的某个人员名下，而是指配到各种角色、部门或工作组上，则该工作流的应用不受企业中人员变动的影响。同样此工作流也可以被应用于不同的场合(应用数据库)，只要配置的组织数据库也定义了这些角色、部门或者工作组。

一个预定义的工作流可以是另一个工作流定义中的组成部分。因此一个常用的预定义的工作流部分可以被组合到用于各种业务的不同工作流之中去。

4. 自由组合多样路径选择

在Espresso工作流系统中，当一个预定义工作流活动完成后，接下来的活动是由从这个活动引出的流线(link)来决定的。一个流线的路径选择(routing options)可以被指定为必经，多向选择，单向选择，条件或者备用。

- 必经(Always)： 一个工作流实例总是从该流线的起点流向终点。
- 多向选择(Multiple Choice)： 一个工作流实例是否经此流线流向终点活动是由完成起点活动的人员选定的。在这个起点活动引出的所有多向选择流线中，可以有多条流线被选中。
- 单向选择(Exclusive Choice)：从此流线起点引出的所有单向选择流线中，必须有一条且只能有一条流线被完成起点活动的人员选定，让一个工作流实例顺此被选流线流向其终点。
- 条件(Condition)：当由LotusScript构成的条件式取值为真时，一个工作流实例从这个流线的起点流向终点。
- 备用(Else)：当一个工作流实例没有经任何其它同一个起点活动引出的流线流向下一个活动时，这个工作流实例就流向备用流线的终点。

一个活动所引出的流线可包含有不同路径选择的流线。用户可以自由地根据实际业务情况定义相应的路径选择。除了必经流线，其它流线的使用是在执行期动态决定的。多向和单向选择的流线由完成活动的人员进行指定。条件和备用流线的选择是由工作流引擎根据工作流实例的运行情况自动决定的。

流线及其路径的定义决定了是否在一个预定义工作流中有并列活动的存在。并列活动的合并需要是由 PAVONE 工作流引擎根据一个工作流实例当时的运行情况自行决定的。

5. 动态任务分配

PAVONE工作流引擎自动按指令分配工作到参与者的任务表中，同时它也可通过电子邮件告知参与者其所应完成的工作。当一个活动完成后，该活动便自动从各参与者的任务表中被清除掉。

预定义工作流中的活动可以指派到活动实例文档的一个数据域。这样在执行期内该活动的参与者便被动态确定。此数据域的内容可以由一个公式给定。如果一个活动被指派到一个角色，通过指定一个数据域来存放角色参数也能动态确定这个活动的参与者。

对预定义工作流的一个活动还可以指定参与者的候补人员。候补人员既可以针对该活动给定，也可以利用组织数据库中的定义。

6. 检验与仿真功能

PAVONE过程建模器特别配备检验和仿真功能。 在应用预定义工作流前利用此检验和仿真功能可以帮助用户提早发现其中存在的问题，如无限循环，资源缺乏，并行活动序列的不平衡，资源利用中的瓶颈问题等等。

利用仿真功能还能帮助用户形象地观察和了解工作流系统将来的运行情况，预测完成一个工作流实例平均需要的时间，各种资源的应用情况及利用率，各项费用等。

7. 按优先级解决活动死锁

对应一个工作流实例(Instance)可能出现几个活动(称为并行活动)同时运行的情况。当出现几个并行活动完成后流向同一个活动(合并活动)的状况时，在该活动被执行前那几个并行活动实例文档必须被合并(join)成一个。这样当一个并行活动的文档流向合并活动时，它会处于等待状态而不能被执行，直到再没有可能其它并行活动文档流向该活动为止。如果两个以上的并行活动互相等待合并，死锁(deadlock)就发生了。在一个具有复杂网络拓扑结构的预定义工作流实例中难免会出现活动死锁的现象。Espresso工作流系统是通过

活动合并优先级来解决死锁问题的。当对应一个工作流实例的所有并行活动都处于等待状态时，其中一个合并优先级最低的活动则被取消等待状态。PAVONE过程建模器会探测出各种可能发生死锁的合并活动组，让用户给组内的活动指定互不相同的合并优先级。

8. 工作流版本管理

Espresso的工作流版本管理功能使得运行中的预定义工作流可以得到修改而不影响系统的正常运行。每一个在PAVONE过程数据库中存放的预定义工作流可以有不同的运行版本。一个运行版本可以同时被若干应用数据库所使用，但是一个应用数据库只能使用每个预定义工作流的一个运行版本。

只要一个预定义工作流的运行版本被联接到一个应用数据库上，在该应用数据库上马上就能创建相应的工作流实例。一旦断开某个运行版本与一个应用数据库联系，在此应用数据库上已经按照该运行版本创建的工作流实例仍然可以继续执行下去。

5.5.3 Espresso 工作流的应用范围

Espresso工作流具有广泛的应用泛围。它可用于工业，银行业，保险业，通讯业，信息技术(IT)服务业，和各事业机关单位。Espresso工作流特别适合于以文件为基础的，需要多人(在不同地点/跨国)合作，并带有机动处理的业务过程。它有助于迅速实现业务过程的自动化。借此企业可以节省成本，便于经营管理，提高工作效率，提供优质服务。

5.6 几类工作流产品发展情况

下面我们将简单分析几类工作流产品的特点和目前的市场发展状况。

1. 以通讯为中心的工作流产品

通讯类的产品代表了工作流产品发展的最新的趋势。它们将工作流技术带入了过去无法应用的领域。该类产品的市场占有率每年以超过 40%的速度增长。通讯类工作流产品已经从最初的解决客户请求（customer inquiries）问题逐渐经过电子商务配置（electronic commerce deployment）过渡到实现用户参与（customer care）。

通讯类产品早期的领先者是 Edify，它的目标主要是电子商业服务，目前它仍占据该类产品市场的绝对优势（51.0%）。PegaSystems 和 TALX 这几年也有了长足的发展，它们的

市场主要在自助式销售业，分别以 38.3%和 10.8%的市场份额位居第二和第三位。Mosaix 在电话业不断扩大的市场使得它的发展令人瞩目。

2. 以文档为中心的工作流产品

该类产品的应用主要集中在电子文档的生成、汇集和全局的管理。这种类型的工作流通常是文档管理应用软件中的一个部分。近几年文档自动化技术的迅速发展和电子文档的逐渐被接受促进了文档类工作流产品的普及——每年以 30%的速度迅速增长。1997 年发展最快的是 Open Text。它的 Livelink Intranet 将文本搜索（text search）、文档管理和用于 web 领域的工作流产品集成在一起，迎合了企业对功能更强的能够平衡各种新的企业应用的工作流的需求。然而 1997 年在所有的文档类产品中最大的赢家还是 Documentum，它以绝对的优势位居榜首。它可以在应用软件原有的环境中定义文档类工作流，因而它可以为企业提供适合自己经营过程的工作流应用产品。

3. 以过程为中心的工作流产品

过程类产品在三类产品中发展时间最长、所占市场份额最大。不同背景的软件供应商都参与了该类产品的开发与应用。我们前面介绍的四个工作流产品都可以算做过程类工作流产品。

在所有的以生产过程为中心的工作流产品的公司中，Staffware是发展最快的。它的增长率连续两年位居首位。1996年它的增长率达到了120%，1997年是76%。它所提供的一套工具可以使用户很容易地将工作流集成到经营过程应用软件中，吸引了很多顾客。另一个在工作流工具开发中新近崛起的是Jetform公司。它在最近几年的业绩也相当引人注目。Jetform在1996年收购了Delrina后，在电子表格领域内占据了主导地位，1997的收入比1996年增长了62%。在以过程为中心的工作流产品中，FileNet以占有16.3%市场的业绩仍然处于领导地位，但最近FileNet已经决定改变企业经营策略，转向相对更有发展潜力的文档类工作流产品。

纵观工作流软件产品由八十年代的萌芽到九十年代的繁荣，我们可以把它总结为三个阶段：第一阶段，主要为应用于某些特定领域的、相对独立的应用系统，比如图像、文档管理系统；第二阶段，主要表现为具有底层的通讯基础结构、能够实现任务协作的应用系统，比如具有消息传递功能的工作流系统；第三阶段，具有图形用户界面的过程定义工具、用户定义与任务执行完全分离的工作流系统，其体系结构基本上符合工作流管理联盟

（WfMC）所提出的标准结构。经历了这三个阶段的发展，工作流产品基本上确定了它在计算机应用软件市场上的独立位置。

第 6 章 工作流模型

6.1 概述

模型是用文字、图表、符号、关系式以及实体模样等描述所认识到的客观对象的一种简化表示形式，它是人们为了研究和解决客观世界中存在的各种问题而对客观现实经过思维抽象后得到的。简单地说，模型就是所描述客观对象的抽象表示。一般来讲，模型都包含一个完整的概念集合、一套相应的表示方法以及必要的规则约束，它们为人们抽象地表达客观对象提供了一个参考性的框架环境。

工作流模型是对工作流的抽象表示，也就是对经营过程的抽象表示。工作流管理建立阶段（Build-Time）的功能主要是完成经营过程的计算机化的定义，也就是完成过程建模的任务。在这个阶段，利用一个或多个建模方法及其相应的建模工具，完成实际的经营过程到计算机可处理的形式化定义的转化。所得到的定义通常可称为过程模型、过程模板、过程元数据或过程定义。由于需要在计算机环境下运行，所以工作流模型不仅仅要让人读懂，更要让计算机能够理解所定义的工作流过程。也正是因为如此，简单的业务过程通过语言或文字就可以表达完全，无需建立明确的模型，而描述企业经营过程的工作流则必须建立相应的工作流模型，才能实现企业业务流程的工作流管理，尤其是对那些实现许多复杂的并行执行流程的业务过程，只有建立它们的计算机化的模型才可能对流程的执行情况进行有效的监控。

工作流模型除了应该支持完整的工作流概念定义，为建模用户提供定义工作流所需要的组件或元素等主要特性外，理想的工作流模型还应该能够清楚地描述任意业务情况下的工作流，适应用户在建模过程中所提出的各种要求，即工作流描述能力要强，可以描述清楚几乎所有的业务流程类型。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，距离这一理想情况尚很遥远。

文献[31]中给出了对概念模型进行评价的四条标准为：

1) 表达能力丰富：模型提供丰富的建模概念，使得由大量约束集合而成的元素能够在模型中被直接表达出来；
2) 容易理解：不论是领域专家还是普通用户，都能够比较容易地理解模型含义；
3) 最小化：模型中的概念不重叠，每一个基本概念都有它特定的意义；

4) 形式化：模型定义能够作为现实对象的形式化描述。

在大多数情况下，概念模型都提供相应的图例表示，因此，我们认为还应该再附加上对图例的要求，如易读性、完整性。

在进入九十年代以后，市场上已经出现了较多的工作流产品，人们开始对工作流模型进行深入的研究。许多工作流模型来自于不同的工作流产品，有些模型则来自于专门的研究项目或者直接由某位学者提出。由于工作流首先必须描述清楚一个经营过程是怎样进行的，因此，许多工作流模型都是从过程的描述入手，比如流程图、状态图、活动网络图、以及 Keller[7]等人提出的 EPCM 模型（事件过程链模型）等等。这一类基于有向图模型的优点是比较直观，容易理解，一般情况下，图中的节点表示过程中的活动或者状态，而有向弧则表示节点间的时序依赖关系。不少工作流产品采用了此种类型的模型，但其缺点是比较简单，不能处理复杂的过程逻辑。

IDEF 系列方法被广泛地用于企业建模和过程建模。它包括一系列的建模方法，包括功能建模方法（IDEF0），信息建模方法（IDEF1/IDEF1x），动态行为建模方法（IDEF2），过程建模方法（IDEF3），面向对象建模方法（IDEF4）等。通常用于过程建模的有 IDEF0 和 IDEF1x，IDEF3 方法等。IDEF0 方法用于描述系统的功能结构，IDEF1x 用于描述信息实体及其信息实体之间的关系，IDEF3 用于描述过程。很多用户直接用 IDEF0 方法进行过程建模，隐含着将 IDEF0 功能模型中的输入/输出关系作为活动的先后顺序关系，而将其表示在图中的功能的左右排列顺序上，事实上 IDEF0 模型是没有这个语义的。IDEF0 模型主要描述的是功能的输入和输出、执行控制和机制，以及功能的递阶分解，它并没有功能执行顺序的定义，因此从原理上，IDEF0 描述了功能的输入、输出和资源的使用情况（如用机制来表示），可以将其作为过程单元的接口表示，但用其进行过程建模从原理上来说是不合适的。IDEF3 则是用两个基本的组织结构——场景描述和对象来获取对过程的描述，相应的有两种描述方式：过程流网 PFN（Process Flow Network）和对象状态转移图 OSTN（Object State Transition Network Diagram）。过程流网是以过程为中心的视图，它注重过程中活动的出现及其次序。IDEF3 用过程流网作为获取、管理和显示以过程为中心的知识的主要工具。在过程流网中包含了不同人员对事件与活动、参与这些事件的对象、以及操纵这些事件的行为之间的约束关系等知识。

STEP Part 49（过程结构和属性）是用中性语言 EXPRESS 定义的，它是 STEP 的一个组成部分，用于产品生产过程计划的描述。它定义了过程（Process）和动作（Action）的描述方法，适用于各种离散过程定义。STEP Part 49 的建模元素包括：过程间关系、过程

效果、过程属性、资源、资源属性、过程表示、资源表示、过程和产品间关系。这些元素组合使用即可完成一个过程计划的定义。Part 49 分为三个模式：方法定义模式、过程属性模式和过程属性表示模式。方法定义模式约定用于修改产品定义与定义产品制造的指令。过程属性模式用于定义过程中涉及的动作、动作的属性、过程执行所需资源的属性、以及过程执行产生的产品属性。这些过程属性定义动作、资源和产品的属性。过程属性表示模式用来定义资源、动作或可能使能一个过程所需的属性，如资源参数、动作参数等。虽然 Part 49 描述能力很强，但由于 STEP 采用了低层术语的表述方式，在作为高层系统建模和设计使用时显得比较繁琐，这也反映了这种方法的抽象表示能力的不足。

Winograd 与 Flores 在语言行为理论的基础上提出了一种基于对话的工作流模型[3, 32, 33]，这种工作流模型从客户方与服务方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，Action 技术公司的工作流产品 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（Information Control Nets）模型[34]，它实际上是高级 Petri 网的一个延申，在这里库所表示活动，而变迁则表示活动间的转移。类似的还有文献[33]中给出的采用有色 Petri 网来描述工作流过程的方法。Van der Aalst 则在 Petri 网的基础上定义了 WF-net[12]，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。另外，还有许多其他形式的工作流模型，比如文献[36]给出了一种基于活动树（Activity Tree）结构的模型，它用一个树状结构来表达工作流过程，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[37]，即代理/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，这样才能够构成一个完整的业务过程模型。因此，人们便在工作流模型中加入了描述数据、组织、资源的相应部分，比如工作流管理联盟（WfMC）就明确提出了工作流相关数据（Workflow Relevant Data）、工作流控制数据（Workflow Control Data）及工作流参与者（Workflow Participant）、角色（Role）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具有一定独立性的模型，这些模型采用了更为细致的

描述方法来描述企业中有关组织、资源和信息的结构。它们辅助过程模型来实现对企业业务过程的全面描述。比较典型的有 WIDE 项目中提出的由组织模型、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，在组织模型与信息模型中分别定义了灵活的组织概念与数据类型来支持企业复杂的人员组织结构和丰富的数据形式。惠普实验室在文献[38]中提出了一种资源模型，它将人员、组织、硬件、软件等各类“资源”纳入了一个层次化的树状框架下。在 MOBILE、DOPAS 原型系统中则提出了动态组织模式的概念[39]，通过组织对象和组织关系这两类基本组件，用户可以定义自己的组织模式。

为了便于交互和在不同格式的模型之间实现转换，有的模型还提出了规范的描述语言，我们称之为“工作流定义语言”。比较典型的有 WfMC 推出的 WPDL（Workflow Process Definition Language），IBM FlowMark 的 FDL（FlowMark Definition Language），METEOR2 项目定义的 WIL（Workflow Intermediate Language）等。这些工作流定义语言都有着自己特定的语法规范，包括标识符、关键字、文法规则等，有的还开发了相应的编译器，用于生成工作流运行的可执行代码。另外，值得一提的是，NIST 单独提出了一种过程描述语言 PSL（Process Specification Language），旨在统一各种与制造业过程定义有关的语言，当然也包括对工作流的定义，但实施效果并不理想。

以下我们将着重介绍几种有代表性的工作流模型，它们都具有比较突出的特点，并代表了一种较为普遍的观点，对于今后开展工作流建模问题的研究有着一定的参考价值。

6.2 基于活动网络的过程模型—FlowMark 工作流模型

FlowMark 是 IBM 公司在 20 世纪 90 年代中期推出的工作流产品，其目标是实现企业的文档路由（document routing）和过程自动化（process automation），这也是传统的工作流产品的典型特点。

在 FlowMark 中，一个完整的经营过程（工作流过程）由一个无自环的有向图构成。有向图中的节点元素表示可执行的步骤或任务，节点间的连接弧代表了过程中的控制流与数据流。组成模型的元素包括过程（Process）、活动（Activity）、模块（Block）、控制连接弧（Control Connector）、数据连接弧（Data Connector）和条件（Conditions）。

1） 过程：由一系列具体的步骤组成，为完成某一预定目标而定义。在这里，一个过程就是用一个有向图来表示的一个工作流，比如，向银行申请贷款就是一个过程。

2） 活动：过程中的每一个步骤是一个活动，在图中由一个节点元素来表示。它可以是程

序活动（Program Activity），也可以是过程活动（Process Activity）。程序活动是指为活动绑定了一段程序代码，当活动开始时，相应的程序就执行；过程活动则是在活动的基础上嵌入了一个过程，当活动开始时，也就是开始了相应的过程，过程活动主要用于子过程的嵌套描述与模型的层次化分解。图 6.1 给出了活动的内部结构，主要包括输入数据箱、输出数据箱、开始与结束条件、状态以及绑定的相应程序或过程，它们的具体特点以及相应的应用场合将在下文中结合模型的其它元素进行描述。

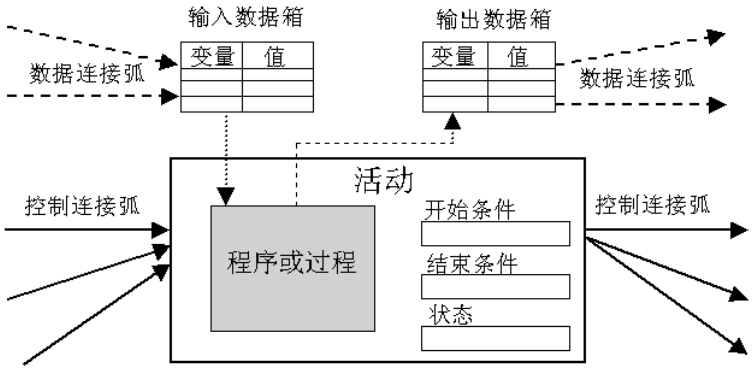


图 6.1 FlowMark 工作流模型中活动的内部结构

3) 模块：有些类似于“过程”，但二者的区别在于，一个过程有一个唯一固定的标识，它可以在不同的工作流过程中使用；而模块则没有这一项，它只能被一个工作流过程所使用。模块也具有输入数据箱与输出数据箱。如果用编程语言来作比喻，模块就好比是介于“Begin”与“End”之间的内部代码，而过程则是外部的连接库代码。

4) 控制连接弧：用来定义两个活动之间的执行顺序，在有向图中以节点间的连接弧表示。当连接弧的起始节点执行完毕，工作流机将会根据控制连接弧的定义进行过程导航，使连接弧的终止节点能够被执行。控制连接弧与它们所连接的活动节点构成了整个工作流过程的控制逻辑。

5) 数据连接弧：定义两个活动间的信息流，在有向图中也是以节点间的连接弧来表示，由前一个活动的输出数据箱指向后一个活动的输入数据箱，意味着前一个活动的输出信息将被后一个活动所使用。数据箱中包含了工作流相关数据以及与该活动执行有关的具体应用数据，如图 6.1 所示。

6) 条件：用于定义过程执行中的约束。主要分为三种基本类型，属于活动外部的有一种，即转移条件；属于活动内部的有两种，即开始条件与结束条件。

（1） 转移条件是定义在控制连接弧上的条件，根据对转移条件的判断结果是“True”

与“False”来决定是否执行下一个活动。

（2） 开始条件定义了活动在什么情况下才能开始执行，如对于用“AND”连接符连接几个前活动的开始条件是要求所有指向该活动的控制连接弧必须全部发生转移后，该活动才能够开始；对于用“OR”连接符连接几个前活动的开始条件是仅要求指向该活动的控制连接弧中有一条发生转移，该活动就可以开始。开始条件的设置提醒我们区分这样两个概念：活动被使能与活动被执行。当满足转移条件，控制连接弧发生转移时，我们称控制连接弧所指向的活动被使能，即该活动有可能被执行；但活动是否真的开始执行，则需要通过活动内部开始条件的判断来决定，只有开始条件得到满足，活动才真正开始执行。

（3） 结束条件定义了活动在什么情况下才能够结束，当活动执行完毕后，活动的结束条件就被检验，如果为“True”，活动则结束；如果为“False”，则活动被重新执行，直到满足结束条件为止。通过结束条件的设置，可以定义需要多次循环执行的活动。

在介绍了上述组成工作流模型的基本元素之后，我们给出一个模型的实例，为简单起见，模型图只画出了控制连接弧，而未给出数据连接弧，如图 6.2 所示。这是一个按客户要求定制生产自行车的工作流过程：首先检查客户定单，获取客户的需求信息；然后并行地开始三个活动，分别是计算货物（这里指自行车）价格、检验库存、检验技术可行性；当这三个活动全部完成后，根据它们的具体执行情况来进行决策；决策的结果有两个，一个是决定生产符合客户要求的自行车，另一个是对客户提出修改需求的建议；若是前者，则开始准备确认信、发送供货单、进行生产与送货；若是后者，则提出修改建议、发送回信。图中的数码 1 到 6 表示工作流实例的一次执行过程，在这个执行过程中，决策活动的结果是决定生产符合客户要求的自行车。

在运行时阶段，与活动相联系的一个重要属性是状态，因为状态是监控过程执行情况的基础。当活动可以开始执行时，状态为“就绪（Ready）”，而且只有处于“就绪”状态的活动才能被执行，执行方式包括自动与人工两种类型；当活动开始执行后，状态由“就绪”转变为“运行（Running）”；当活动执行完毕而且满足了活动的终止条件，状态则由“运行”转变为“终止（Terminated）”。当然，还有其他的状态与活动相联，比如异常挂起等。在正常的执行情况下，即不发生异常与例外的情况下，过程的推进主要是在以上提到的三个状态之间进行转换。

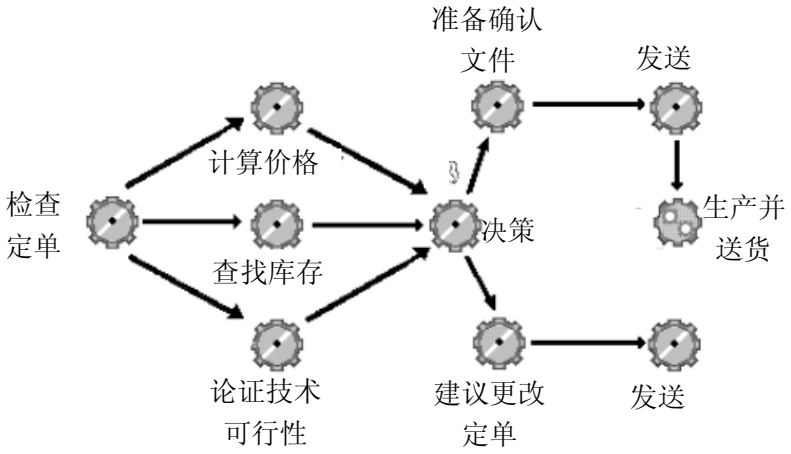


图 6.2 一个定制生产自行车的工作流过程

当一个过程开始以后，没有输入控制连接弧的活动（它们是初始活动）的状态将被首先置为“就绪”，因为它们将最先被执行，而且它们的执行表示了一个业务过程或者一个工作流实例执行过程的开始。任何一个活动执行完毕后，需要判断其终止条件是否满足，若条件不满足，则该活动由“运行”状态被重新置为“就绪”状态，以便继续执行；人工终止条件满足，则活动被置为“终止”状态，继而工作流机对由该活动发出的所有控制连接弧上的转移条件进行判断，以决定控制连接弧是否能够发生转移。对于转移条件判定为“False”的控制连接弧，它所指向的后继活动的状态被置为“终止”（这些活动并未经历“就绪”与“运行”状态，而是直接被标定为“终止”，原因是在这样的条件下它们不可能被执行），然后由这些被置为“终止”的活动所发出的控制连接弧也被置为“False”，接下去重复同样的操作，把相应的后继活动置为“终止”，把“终止”的活动所发出的控制连接弧置为“False”，这样的一个递归过程被称为“无效路径削减”。对于能够发生转移的控制连接弧，则继续判断该弧所指向的活动的开始条件是否满足，若满足，活动的状态则被置为“就绪”，以等待执行；若不满足开始条件，且活动不可能被执行，则该活动的状态被置为“终止”，然后进行前面提到的“无效路径削减”过程。当一个过程结束时，它其中的所有活动都应该处于“终止”状态。不难想到，其中的一些活动是真正被执行过而终止的，而有一些则是在“无效路径削减”的过程中被置为“终止”的，实际上这一部分活动并未真正运行。比如在上图给出的示例中，如果 1—6 是过程的执行顺序，那么“提出修改建议”与“发出回信”就是两个被“无效路径削减”过程所削减的活动。

上述的推进过程可以表示为图 6.3 所示的状态转移图。

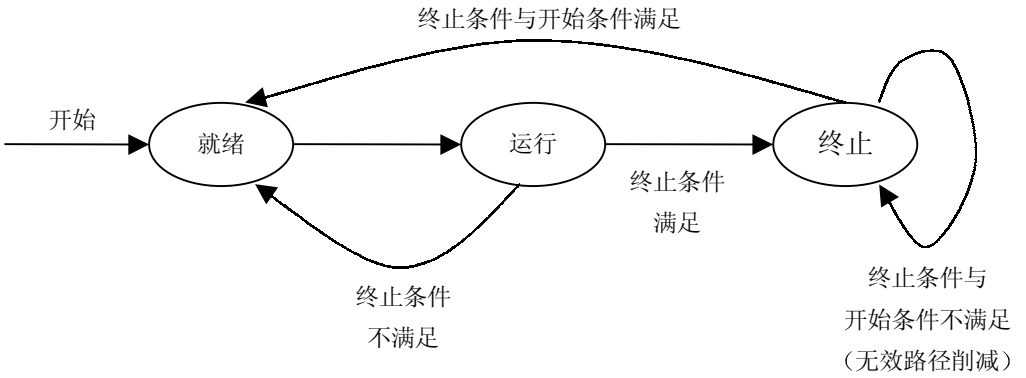


图 6.3 活动的状态转移图

FlowMark 的工作流模型是一种典型的基于活动的 IPO（输入－处理－输出）模型，以活动作为构成过程的基本单元，以连接弧体现过程逻辑，活动的输入数据箱与输出数据箱成为活动输入输出的接口，再辅以条件的设置，就能够比较简便、灵活地实现企业经营过程的建模。

从模型的语义上，FlowMark 明确地将控制流与数据流进行了分离，这样，一个工作流过程可以被看成是由控制流图与数据流图叠加而成的。FlowMark 的这一特点，将会带来如下好处：

（1）从系统分析的角度来看，有利于通过过程模型来提取建立功能视图与信息视图模型所需要的相关信息，便于对企业这一复杂系统进行深入地分析，比如从过程模型中提取信息并输入到 IDEF0、IDEF1X 建模工具中，从而辅助功能模型与信息模型的建立；

（2）从系统实现的角度来看，这种分离直接对应于控制流管理与数据流管理的相互分离，从而实现两种不同性质的流管理上的独立。在 FlowMark 的系统实现上，IBM Exotica/FMQM 实现了分布式工作流的控制流管理，而相应地，基于 Lotus Notes 的文档系统则实现了分布式工作流的数据流管理。在这样的分布式工作流执行环境中，逻辑上控制节点与数据节点（在物理上二者可能位于同一个内存空间中）的分离将带来如下好处：①有利于保护原有的数据系统，不必强迫企业改变原有的数据管理方案；②有利于数据节点能够自主地、柔性地、高性能地处理复杂结构的数据而不会影响整个工作流系统的效率，特别是对多媒体数据的处理，这是唯一可行的方法；③有利于增强系统的可扩展性、可靠性以及容错性等分布式性能。

除了 IBM 的 FlowMark 所提出的这种基于活动网络的工作流模型以外，还有其他的工作流产品（如 InConcert）或原型系统（如 METEOR2）也推出了类似的模型。这些模型在宏观含义上是一致的，都明确地反映出基于活动网络的过程模型所应该具有的特点，而仅

仅在细微之处有所不同，比如模型元素的类型、模型元素的语义等等。总的来说，这一类模型所共有的特点是简单、直观、便于理解，适合于流程较为固定、异常情况较少的生产型工作流的建立，有利于企业规范自身的内部流程；但这类模型往往缺乏柔性，对紧急情况的应变能力不足，显得比较死板。

6.3 事件驱动的过程链模型（EPC）

一种应用比较广泛、用来描述企业事件与经营过程的传统方法就是事件驱动的过程链模型（Event-driven Process Chain），简称为 EPC 模型。它主要被用于企业的经营过程重组（BPR）、工作流的定义与控制、软件的配置与开发、基于活动的成本（ABC）分析以及符合 ISO 900x 认证标准的质量文档的规范。世界范围内已经有数千家公司、企业使用 EPC 对自己的经营过程建立了模型，一些软件供应商（如 SAP）还提供了相应的软件工具来支持 EPC 的建模过程，这些软件工具通常都提供一些标准的通用 EPC 模型用以支持企业用户对过程的定制。

EPC 是由 Keller 提出的。EPC 的主要元素就是功能和事件：功能被事件触发，功能也能产生相应的事件。经营过程的控制流就这样由交替出现的功能和事件彼此连接而构成，控制流的分支选择、汇合连接以及并发进行则通过逻辑操作符（比如与、或、异或）或者更复杂的表达式来完成。图 6.4 给出了一个一般形式的 EPC 过程模型，直观地展示了模型中不同元素之间的相互连接关系。

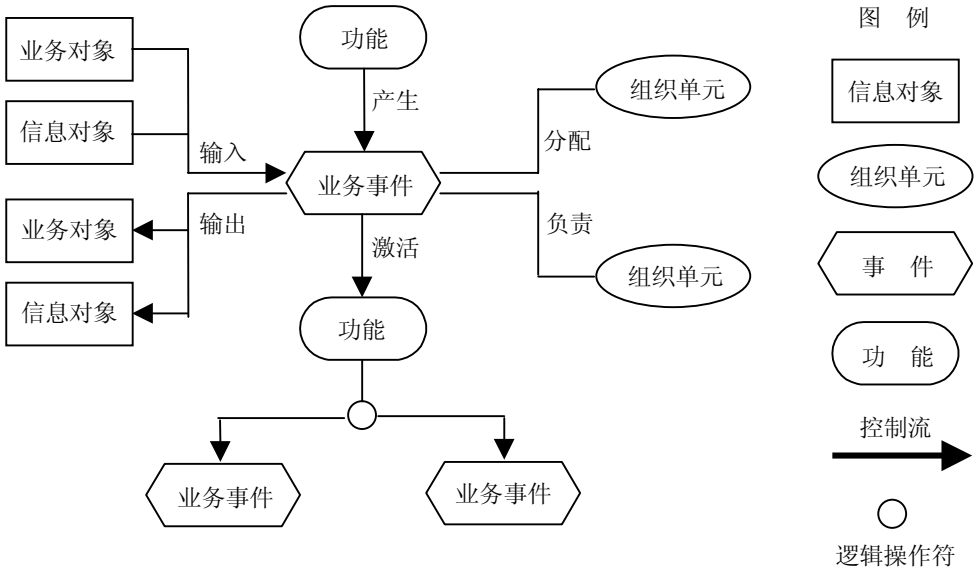


图 6.4 EPC 元模型

对于图 6.4 中所给出的各类模型元素，其描述如下：

1) 业务事件：当过程的状态发生改变时即产生业务事件，通常是对完成某一功能而做出的响应。

2) 业务功能：通常是一个活动或者一项任务，由组织单元中负责此功能的人来完成；当然在工作流系统中也可能通过激活应用来自动来完成。

3) 控制流：连接功能与事件的有向弧，用以表示过程的控制逻辑。

4) 逻辑操作符：用来实现控制流的分支与汇合，主要包括与、或、异或三类简单的逻辑操作。

5) 信息对象：完成功能时所需要的数据信息，它们既可以作为功能的输入，也可以是功能输出的结果。类似于工作流管理联盟所定义的工作流相关数据。

6) 组织单元：负责执行功能单元的组织。

利用 EPC 元模型所给出的不同元素，我们建立了一个商务旅行的过程模型，如图 6.5 所示。申请人需要首先填写表格，然后经过上级主管或者经理（二者选其一）批准同意即可。过程中用到的信息对象包括旅行申请表和同意批复，组织单元包括上级主管和经理。

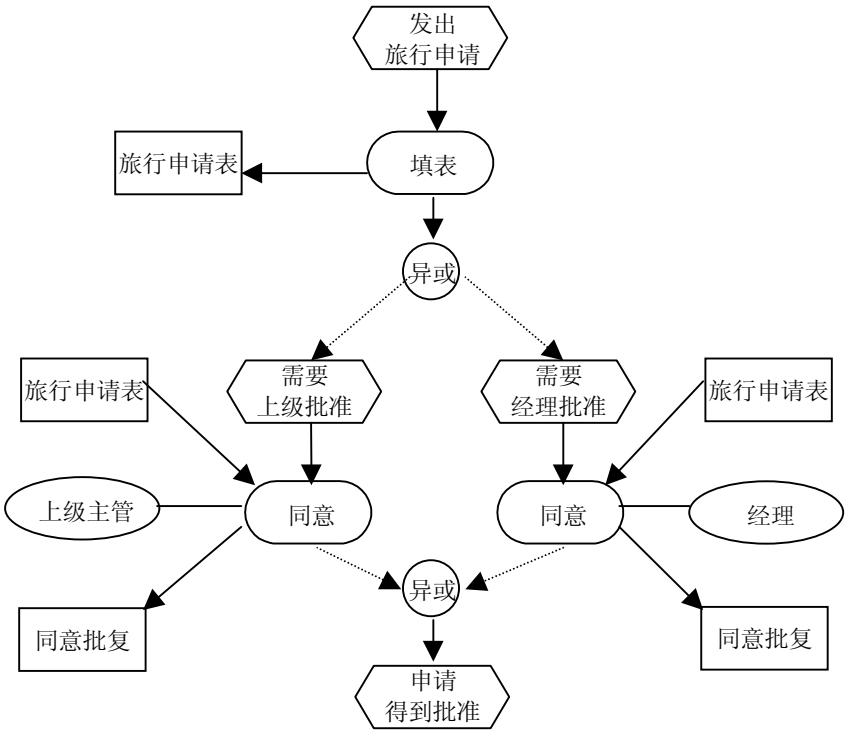


图 6.5 商务旅行的 EPC 模型

EPC 模型的一个很大的优点就在于它兼顾了模型描述能力强与模型易读性这两个方面。因此，EPC 经常被用于在与未受过专业建模训练的普通用户讨论经营过程的场合；同

时，EPC 模型经过改进、提炼后，也同样可以作为一个企业信息系统的需求定义。这也正是许多企业与公司利用 EPC 来进行过程建模的原因。为了进一步提高建模的质量与效率，不少研究人员试图把已有的建模方法（如 E-R 图、面向对象方法）与 EPC 相结合，成为一种集成的建模方法，能够更有针对性地面向某一领域，如信息系统开发、企业经营过程建模等。Thomas Allweyer 就把 EPC 与 UML 相结合，用于面向对象的经营过程建模当中。

6.4 基于语言行为理论的工作流模型——ActionWorkflow

一百年前，Frederick Taylor 在《科学管理》一书中详细阐述了科学管理的主要原则，他把一个组织的工作描述成为一系列的任务，每一个任务都是工人们具体的、严谨的活动过程；而管理就是使这些任务在一定的计划下以最优的方式进行。这种管理的概念一直被延续下来，即使是在计算机出现并发展了多年的今天，我们依然能够找到它的影子——在许多计算机管理系统中，人被当做了一种功能实体而建立模型，能够对输入进行处理并产生相应的输出。这也造成了人们普遍认同的一种对企业经营过程的看法——IPO 模型：即过程中的每一个基本活动都由输入（I）、处理（P）、输出（O）这三部分组成。尽管 IPO 模型在二十世纪大规模的生产活动中得到了广泛的应用并取得了令人满意的成果，但正如 Flores 所说，在以遍布全球的互连网络为通讯基础的今天，人们必须要重新考虑对“工作”这一概念的理解问题以便能实现更为有效的组织管理，而有效的行为协作不失为一种好的观念模式，因为通过透视协作联系中的“语言行为（Speech Acts）”就可以监控工作的进展情况。Flores 认为，许多在管理上的失败教训都来自于对工作本质的错误认识。文献[42]中也指出，以传统的提高产量的管理方式来对待知识型工作并不见效，他强调在企业人员之间要加强有效的信息联系与协作。

IPO 模型对于观察信息与物料的流动过程比较合适，但不利于观察不同角色之间的委托与承担行为，在对客户提出需求、服务方与之答成协议这一过程上则完全是盲点，而且从模型语义上不提供保证客户满意的机制。在一个组织中，过程通常可以分为三类：物料、信息、人的协作。前两类属于传统的 IPO 型，它们主要处理对象（物料、数据等）的移动问题，把这些对象送到对它们进行操作、转换、消耗、或结合更新的地点。人的协作过程则主要处理对工作提出需求，然后在做什么、谁去做、何时做、是否满意等问题上答成协议。这类过程与前两类过程的不同是信息与物料的移动是工作产生的结果而并非是工作本身。

Winograd 与 Flores 在“语言行为”理论的基础上提出了一种协作过程的建模方法。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——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。如图 6.6 给出的一个对话状态图，就是一个发生在 A、B 之间的协作过程，沿着粗线方向前进，整个过程将成功地结束，最终的状态是“5”，而细线则表示这一过程中可能出现的重复与例外情况，可作为过程终止的状态是“5”、“7”、“8”、“9”，它们以加粗边缘表示。

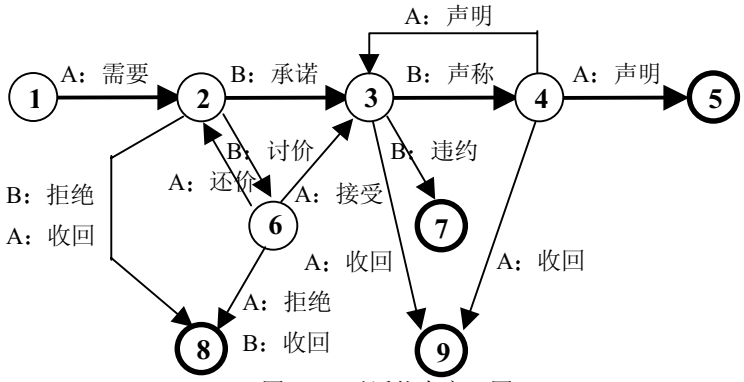


图 6.6 对话状态交互图

基于这一对话状态图，Winograd 与 Flores 建立了协作过程模型。协作过程的基本组成元素是一个闭合的环，它连接了客户方与服务方两类角色，类似于图 6.6 中的 A 与 B。前者向后者提出要求，后者则尽力使前者满意。如图 6.7 所示，这个闭合的环被四个语言行为（Speech Acts）分为四个阶段，实际对应了图 6.6 中水平粗线所经历的“正常”路径：①需求阶段：客户方向服务方提出具体的任务要求，或者说客户方接受服务方的服务请求。在这一阶段，客户方会说：“我需要”；②协商阶段：双方针对客户的满意条件进行商讨，最终由服务方对这些条件做出承诺。在这一阶段，服务方会说：“我保证”；③执行阶段：服务方执行任务直至最终通知客户该任务已经完成。在这一阶段，服务方会说：“我已完成”；④满意阶段：客户方收到服务结果并声明自己满意。在这一阶段，客户方会说：“我满意”；整个过程结束。

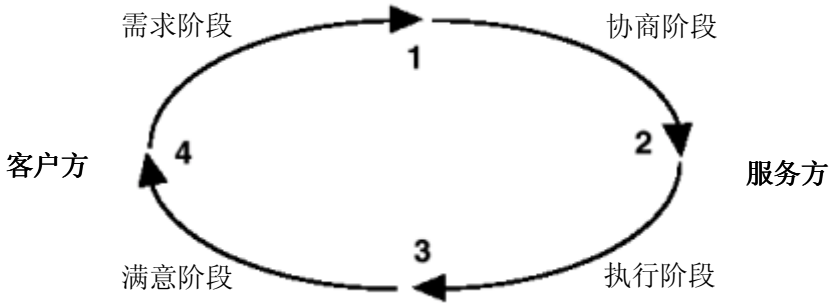


图 6.7 一个工作流环

在服务方执行任务的过程中，它还可以向其他人发出新的请求以帮助自己完成对客户方的承诺，这样原来的服务方就变成了其他人的客户方，其他人将为这一新的客户方提供服务，从而形成了一个新的闭环，这个闭环与前面的闭环是相互连接的。如此延拓下去，多个担任不同角色的人将被涉及进来而为最初的客户提供服务，形成了一个由多个闭环互相连接而成的网络。如图 6.8 所示：

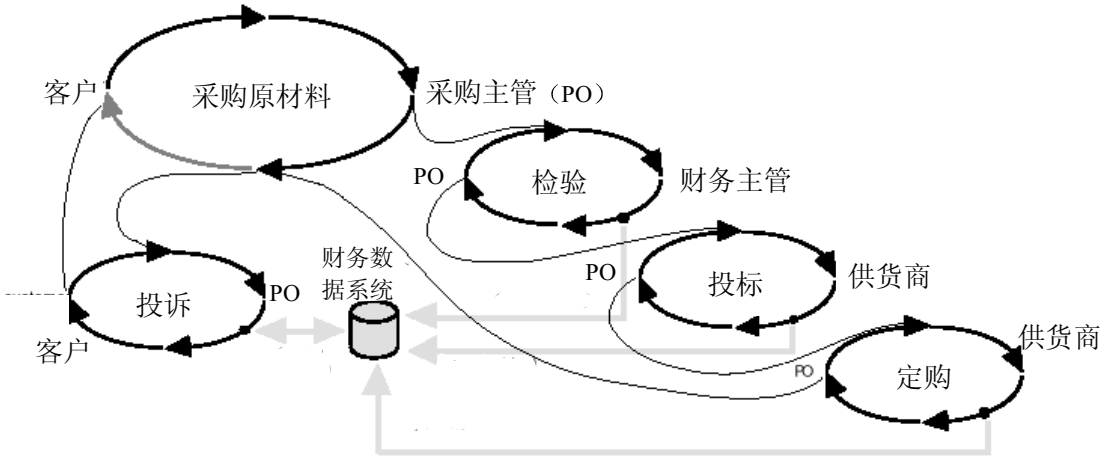


图 6.8　一个定货的工作流过程

图 6.8 给出了一个采购原材料的经营过程，它包括一个主闭环（采购原材料）与三个从属闭环（检验、投标、定购）。在主闭环中，采购主管是服务方；而在从属闭环中，采购主管则成为了客户方。所有的闭环过程都以同一个财务数据系统作为信息源。我们把这个过程大致描述如下：首先由客户（比如企业内部需要某种原材料的一个生产部门）向采购主管提出原料的采购要求，在双方协商后，采购主管开始执行采购任务；采购任务的执行需要经历串行的三个步骤，分别是检验库存、投标、定购。在这三个步骤中，采购主管分别与财务主管和供货商打交道，直到最后完成采购的任务。

在许多采购部门里，主闭环（采购原材料）中的第四阶段常常被省略（如图中所画的虚线部分），这就意味着采购部门并不去查看所购原材料是否按时到货以及需要它的客户是否满意。这样，往往导致不满意的客户来投诉，使得采购部门不得不开始一个新的闭环（解决投诉）来结束整个采购过程，在时间、精力与人际关系方面都是一种不必要的浪费。上图揭示了一个在设计经营过程方面需要注意的问题，那就是不完整的工作流程一定会产生问题，如果问题持续下去而得不到解决，必然会引起报怨而妨碍实现经营过程的永久不变的最终目标——使客户满意。

多个闭环之间执行的逻辑顺序是从连接弧上反映出来的，每一条连接弧起始于前一个闭环某一阶段 X 的结束而终止于后一个闭环某一阶段 Y 的开始。这表明后一个闭环的 Y

阶段将在前一个闭环的 X 阶段结束后开始。至于更为复杂的逻辑关系，可以通过添加其他模型元素及语义来实现，如分支、汇合等。

这种基于语言行为（Speech Act）的过程建模方法被 Action 技术公司所采用并开发了相应的工作流产品——ActionWorkflow。George Mason 大学则利用这一方法改进了安排课程表的流程——这是一个影响到学校中所有学生的复杂的协作过程。

这种建模方法在某些方面也遭到了批评，比如支持层次化建模的能力不足、不适合于比较固定的企业经营过程、建模人员很难完整明确地列出双方所有可能的语言行为等。但在处理以人的交互为特征的经营过程时，此种方法的确显示出优于传统 IPO 方法的能力。表 6.1 给出了基于语言行为的过程建模方法与 IPO 式的过程建模方法在多个不同的方面的比较结果。需要说明的是，这些比较是建立在主观经验的基础之上的，而并不存在客观标准；而且，下面的比较主要是侧重于对经营过程的建模以及经营过程的改进方面，并不侧重于在工作流执行方面。

表 6.1 IPO 模型与语言行为模型之间的比较

比较内容	IPO 方法	语言行为方法
适合的环境	比较稳定的生产活动	临时性、易变化、不固定的经营过程
过程目标	生产产品	实现需求、完成服务
面向的方向	面向市场	面向客户
客户的所得	获得过程的输出	提出的需求被满足
过程边界	由第一个活动直到最后的过程输出	由客户提出需求直到客户满意
模型结构	串行与并行的活动	由多个闭环相互连接的网络
过程内部的连接元素	活动的输入与输出	客户方与服务方两类角色
任务本质	处理物料与信息	实现经营过程的最终目标
建模分析时的侧重点	对过程结构本身的分析	对两类角色之间所做交互的分析
关键指标	过程周期、次品率等标准化参数	客户满意程度

6.5 基于 Petri 网的工作流模型——WF-net

Petri 网作为一种图形化和数学化的建模工具，自六十年代由德国学者 C. A. Petri 提出以来，经过三十多年的发展，已被广泛应用于各个领域进行系统的建模、分析和控制。如通讯协议的验证、网络性能的分析、并行程序的设计、柔性制造系统的控制、知识推理以及人工神经元网络等。

Petri 网是一种适用于多种系统的图形化、数学化建模工具，为描述和研究具有并行、异步、分布式和随机性等特征的复杂系统提供了强有力的手段。作为一种图形化工具，可以把 Petri 网看作与数据流图和网络相似的通讯辅助方法；作为一种数学化工具，它可以用来建立状态方程、代数方程和其它描述系统行为的数学模型。

在建模过程中，如果使用条件和事件的概念，那么库所就代表条件，变迁则代表事件。一个变迁（事件）有一定数量的输入和输出库所，分别代表事件的前置条件和后置条件。库所中的托肯代表可以使用的资源或数据。

图 6.9 表示的是四季交替的过程的 Petri 网模型。这是一个条件/事件系统，其中的条件是：“现在是春天”、“现在是夏天”、“现在是秋天”和“现在是冬天”，用圆圈（库所）表示，而其中的小黑点（托肯）表示该条件成立。该系统中的事件有：“夏天开始”、“秋天开始”、“冬天开始”和“春天开始”，用方框（变迁）表示。当某个事件的前置条件成立时，该事件发生，结果是其前置条件被置为不成立状态，而后置事件被置为成立状态，在图上的操作就是把小黑点从一个圆圈移动到下一个圆圈。于是，在任一时刻，小黑点的分布状态就表达了系统的当前状态，系统的演进过程就在图上清晰地表现出来了。

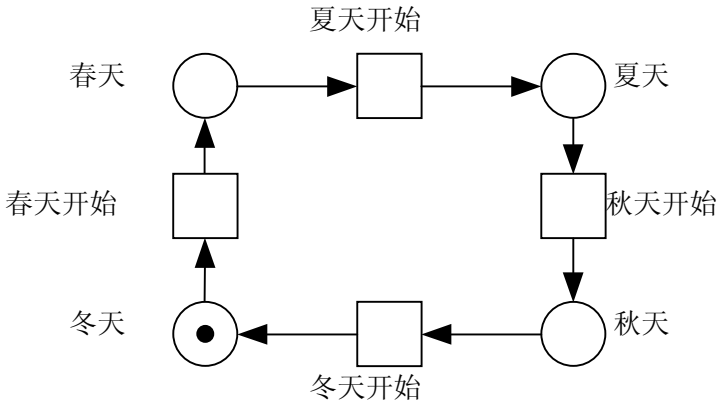


图 6.9　四季交替的 Petri 网模型

随着企业建模、工作流概念的出现以及相关技术的发展，Petri 网也被应用于这一领域。利用 Petri 网进行工作流过程建模主要有以下几个优点：

1)　Petri 网兼顾了严格语义与图形语言两个方面

由于经典 Petri 网（如库所/变迁网）以及高级 Petri 网（如有色 Petri 网、时间 Petri 网、层次 Petri 网、谓词/变迁网）的所有元素都是经过严格定义的，具有规范的模型语义，因此，基于 Petri 网表示的工作流过程也具有十分清晰与严格的定义。Petri 网具有足够丰富的表达能力，完全支持 WfMC 所定义的六种工作流原语，如图 6.10 所示，这些原语被无歧

义地映射成 Petri 网的表达形式。图中以变迁表示实际的活动，而库所则表示活动发生的因果关系，即条件。

在具有丰富而严格的模型语义的同时，Petri 网又是一种图形化语言，具有直观与易懂的特点，使得建模人员能够比较方便地针对模型的含义与最终用户进行交流，以便准确的描述用户环境及改进模型。

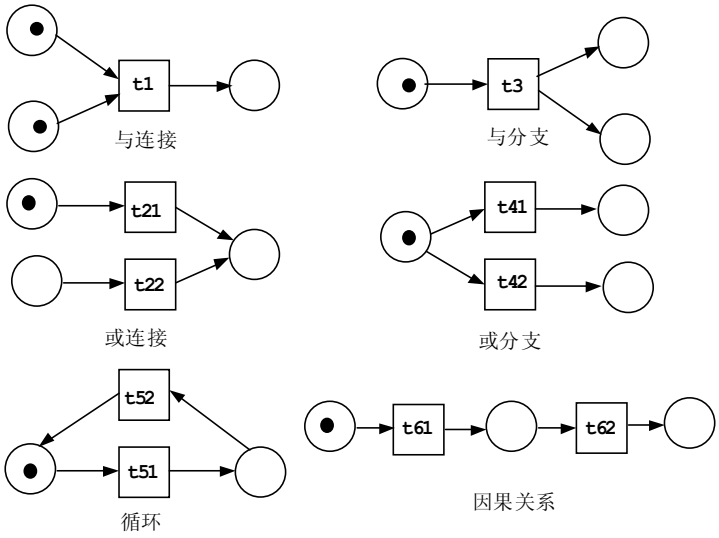


图 6.10　工作流原语向 Petri 网的转换

2)　Petri 网是一种基于状态的建模方法

在许多工作流管理系统中，所采用的过程建模方法都是基于事件的，比如最常见的活动网络法。这类方法把活动与活动之间的转移定义得十分清楚，而对于活动的状态则没有在模型中得以明确的体现。相比之下，Petri 网则是一种基于状态的建模方法，它明确定义了模型元素的状态，而且它的演进过程也是受状态驱动的。使用基于状态的建模方法进行工作流过程定义在以下几个方面要优于基于事件的过程定义：

（1）基于状态的过程定义严格地区分了活动的使能与活动的执行

活动的使能是指活动已经被允许执行，但是不一定立刻就开始执行，在使能与执行这两个状态之间还存在着时间上与条件上的差别。在模型语义上严格地区分这两种状态是十分必要的。Petri 网通过含有托肯的库所来使能相应的变迁（活动），通过变迁的触发来表示活动的执行，从而明确区分了这两种不同的状态；而对于那些基于事件的建模方法，则不能很直观地借助模型自身的语义来对此做出区分，必须附加其它的说明。

（2）基于状态的过程定义具有更丰富的表达能力

由于对不同状态的明确区分，使得基于状态的过程定义在表达能力上具有更大的潜力。举例来说，对于下面的这种情况，许多工作流管理系统的建模工具是无法处理的，如图 6.11

所示：

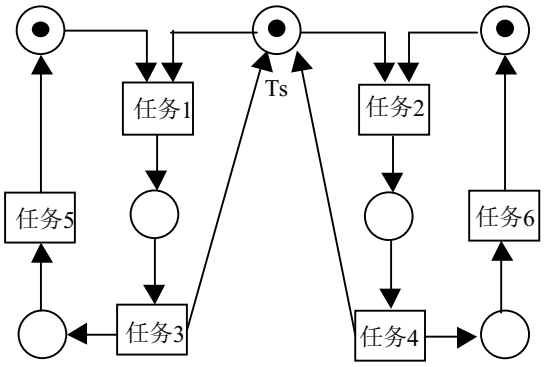


图 6.11 一个具有并发活动的例子

任务 1 与任务 2 是两个并发的活动，即两个活动同时被使能，但是只有其中的一个能够真正被执行。Petri 网通过一个共享的托肯 Ts 准确地描述了这种情况：托肯 Ts 使两个活动任务 1 与任务 2 都可以被触发（执行），但是一旦其中的一个活动被触发（执行），则这一共享的托肯就被转移，立刻使另外一个活动被禁止。

（3）基于状态的过程定义具有更多的柔性特征

对于工作流管理系统而言，具备一定的柔性是必不可少的。比如，能够动态地修改过程实例、可以实现与其他工作流系统的互操作、对异常情况做出响应等等。对于 Petri 网这种基于状态的过程定义而言，只需要对网中的托肯与触发做相应的处理，就能够比较容易的实现上述功能。

3) Petri 网具有强有力的分析技术与手段

经过三十多年的发展，Petri 网拥有了多种可以利用的分析技术，这些技术可以用来分析模型的各种特性，如有界性（安全性）、活性（无死锁）、不变量等；也可以用来计算模型的各种性能指标，如响应时间、等待时间、占有率等。这些分析技术同样也为基于 Petri 网的工作流模型提供了强有力的分析手段，可以用来对工作流过程的一些基本要求进行验证，比如，工作流过程能否最终结束，过程中是否有永远也不可到达的活动等等。利用一些成熟的 Petri 网的仿真工具，还可以对同一过程的不同 Petri 网模型进行评价以便选择性能最优的一个来运行。

综上所述，Petri 网是一种具有较好基础的、适合于描述工作流过程的建模方法。

6.5.1 工作流网的定义

在 Petri 网的基础上，Aalst 提出了工作流网（WF-net）的概念，其定义如下：

一个 Petri 网 PN=（P，T，F）被称为工作流网，当且仅当它满足下面的两个条件——

（1）PN 有两个特殊的库所：i 和 o。库所 i 是一个起始库所，即 $\bullet i = \Phi$；库所 o 是一个终止库所，即 $o \bullet = \Phi$。

（2）如果在 PN 中加入一个新的变迁 t^*，使 t^* 连接库所 o 与 i，即 $\bullet t^* = \{o\}$，$t^* \bullet = \{i\}$，这时所得到的 PN 是强连接的。

从以上这两个对 Petri 网的约束条件不难看出：条件 1 是使工作流网必须具有一个起始点和一个终止点，进入起始库所的托肯代表着一个过程实例的开始，而进入终止库所的托肯则意味着一个过程实例的结束；条件 2 使得工作流网中不存在处于孤立状态的活动与条件，（所谓孤立状态，是指经过该变迁或库所不存在由 i 到 o 的通路），所有的活动与条件都位于由起始点到终止点的通路上。

在工作流网中，库所对应着过程中的条件，变迁对应着过程中的可执行活动，库所中的托肯代表一个过程实例的状态。另外需要说明一点的是，上面对工作流网的定义是最基本的，即使满足上述的两个条件，仍然会定义出具有死锁可能性的工作流过程。

6.5.2 工作流网的基本组件

在库所与变迁的基础上，为了定义出串行、并行、条件选择、循环等常见的过程逻辑，工作流网构造了一些结构化的组件来实现这些功能，在建模时用户可以直接使用这些组件，从而加快用户的建模过程。

（1）串行组件

串行组件用来定义一系列按固定顺序串行执行的活动，它由一条不分支的通路构成，如图 6.12 所示，A、B、C 是三个串行的活动：B 必须在 A 执行完毕后才能执行，C 必须在 B 执行完毕后才能执行。库所 c2 定义了 A 与 B 之间的因果关系，库所 c3 定义了 B 与 C 之间的因果关系。

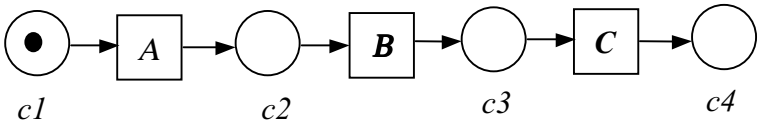


图 6.12 串行组件

（2）并行组件

并行组件用于定义没有严格执行顺序的、可同时进行的分支活动。它需要用到两个基本的工作流执行原语：“与分支”（AND-split）和“与连接”（AND-join）。如图 6.13 所示。

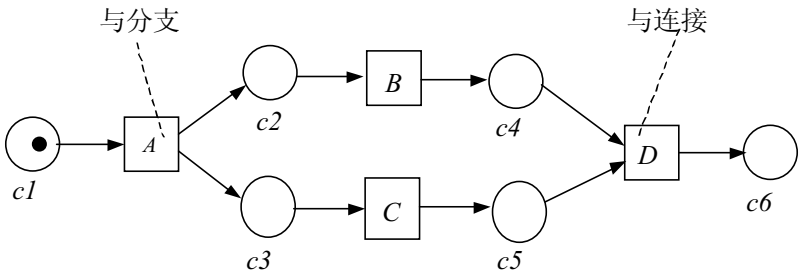


图 6.13 并行组件

图中活动 A 的执行使得 c1 库所中的托肯转移到了库所 c2 与 c3 中，因而活动 B 与活动 C 同时被使能，而且由于 B 与 C 分别位于两个互不相关的支路上，二者之间没有任何相互制约的关系，所以活动 B 与活动 C 能够以任意的顺序执行，可以同时开始，也可以有先有后。在 B 与 C 执行完毕后，活动 D 用来同步这两个支路，以保证 B 与 C 都被完成之后才继续向前推进流程。

（3）条件选择组件

条件选择组件用来定义彼此之间具有相互制约与排斥关系的分支活动，这类分支活动往往是根据具体的执行情况来从中进行“多选一”或“多选多”。条件选择组件也需要用到两个基本的工作流执行原语：“或分支”（OR-split）和“或连接”（OR-join）。但是，在这里需要区分两种不同情况的“或分支”，分别称为“隐式或分支”和“显式或分支”。

- 具有“隐式或分支”的条件选择组件

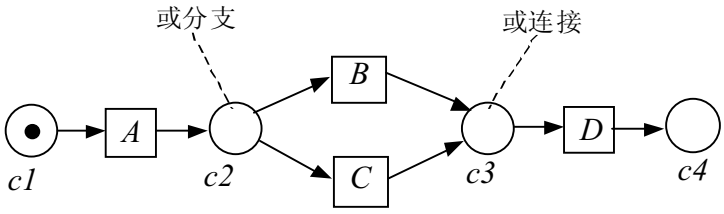


图 6.14 隐式或分支的条件选择组件

图 6.14 中，活动 A 执行完毕之后，把 c1 库所中的托肯转移到了 c2 库所中，由于仅有一个托肯，所以活动 B 与活动 C 只能有一个被执行，这是一个“二选一”的情况。当 B 或者 C 执行完毕后，将托肯转移到 c3 库所中，使后继活动 D 能够继续执行。

当 c2 中含有托肯后，就出现了活动 B 与活动 C 同时被使能的情况，但是究竟哪一个能够被执行，从图中并不能知道（因而被称为“隐式或分支”），这取决于 B 与 C 的触发结果。如果 B 先于 C 被触发，则 B 执行而 C 被禁止；反之则 C 执行而 B 被禁止。在这里 B 与 C 的选择结果与活动 A 无关。

- 具有“显式或分支”的条件选择组件

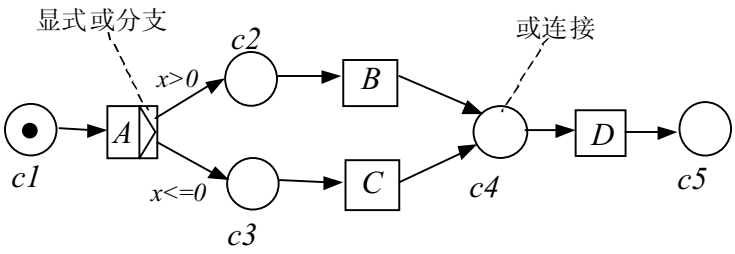


图 6.15 显式或分支的条件选择组件

图 6.15 中，活动 A 具有两个输出库所 c2 与 c3，但是与通常情况下的“与分支”有所不同，在这里 A 表示了一个“或分支”，活动 A 只能根据其某个活动属性 x 的值来决定向哪个库所中输出托肯：当 x>0 时，c2 获得托肯，因而活动 B 将执行；当 x<=0 时，c3 获得托肯，活动 C 将执行。由于从图中可以明确地看出分支中的哪一个活动将被执行，因而被称为“显式或分支”。在这里 B 与 C 的选择结果是由活动 A 决定的，也就是说，当 A 执行完毕后，B 与 C 哪一个将被执行也就确定了。

由于引入了“显示或分支”，为了同“与分支”有所区分，将“与分支”改为图 6.16 所示的表示形式。

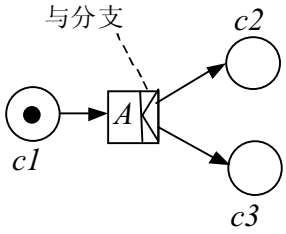


图 6.16 与分支的新符号

（4）循环组件

循环组件用来定义需要重复执行多次的活动，它用到了一个“显式或分支”的执行原语，如图 6.17 所示。

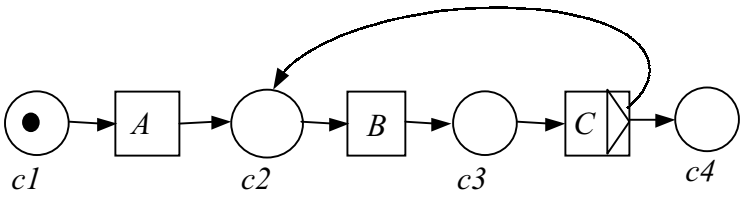


图 6.17 反复组件

在图 6.17 中，B 是被反复执行的活动，而 C 可以理解为一个起控制作用的任务，用来检验 B 的执行结果，以决定是把托肯移到 c4 还是移回 c2。如果托肯被移到 c4，则 B 不再被执行，而是继续推进流程进度；如果托肯被移回 c2，则 B 将被重复执行。一般来讲，循环并非是一种最优的执行逻辑，因为对于整个过程来讲，重复执行同一个任务对于过程的

进展是毫无贡献的。然而在实际情况中，循环却又是不能被忽略的，比如，由客户填写的表格不完整或不正确，在经过有关人员的检查后，需要客户重新填写，这就是一个常见的循环过程。

6.5.3 触发机制

工作流网准确地区分了活动的使能与活动的执行两种状态，对于被使能的活动，它要真正被执行，必须具备相应的触发机制。触发机制可以理解为一种使被使能的活动进入执行状态的外部条件。通常可分为四种类型，它们的记号如图 6.18 所示。

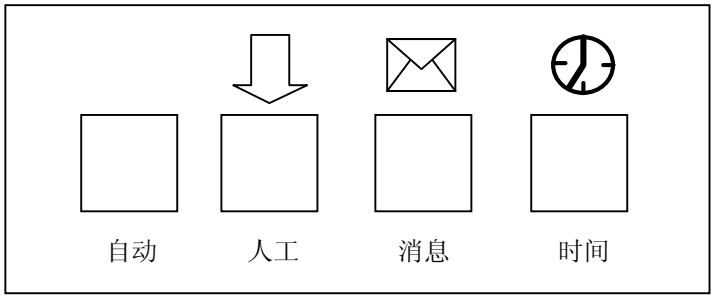


图 6.18 触发机制示意图

① 自动触发：活动被使能的同时就被触发。这种机制一般用于那些通过应用程序来自动执行、不需要与人进行交互的自动型活动，这类活动一旦被使能，就开始执行。

② 人工触发：活动的执行是通过执行者从工作流任务管理器提供的工作流任务表中选择工作项来进行触发。在工作流管理系统中，每个活动执行者都有一个自己的工作流任务表，表中列出了该执行者可以执行（已被使能）的活动实例，当执行者选中某一工作项去执行时，该活动就被触发。

③ 消息触发：由来自于系统外部的消息（事件）来触发活动的执行。比如电话、传真、E-mail 的到达。

④ 时间触发：由控制时间的定时器来触发使能的活动。这对于那些需要在预定的时间或给定时间间隔要求来执行的活动是必不可缺少的。比如某一结算活动必须在下午五点钟才可以开始执行，又比如在温室控制系统中，启动加热器的动作必须是在加热器上次停止加热工作的 10 分钟后才能够执行。

这四种触发机制将被用于工作流网的定义之中，在每一个活动（变迁）的上方，都会标有相应的记号，以指明该活动是通过哪种触发机制来执行的。

6.5.4 一个工作流网模型的例子

本节我们给出一个用工作流网定义工作流过程的例子，例子描述的是一个处理客户投诉的工作流过程。基本流程如下：在公司接到客户投诉时，首先对投诉内容进行登记，然后向客户寄出调查表，同时对该投诉进行评估。如果投诉的客户在两个星期之内返回了调查表，则调查表就被处理；如果超过这一期限，则调查表就被废弃。在对投诉进行评估的基础之上，根据评估意见来决定是否对该投诉进行处理。但是不论是否决定对该投诉进行处理，下一步的活动都要等到客户寄回调查表或者两个星期的期限已满才开始。如果决定处理投诉，则由相关人员执行该任务，处理完后，要经过检查人员的检验，检验不合格，还要重新处理，直到合格后，才把相应的投诉记录归档保存；如果决定不处理该投诉，则直接归档保存，整个过程结束。

根据工作流网的基本定义，通过使用不同类型的基本组件及触发机制，对这个投诉过程进行建模，得到如图 6.19 所示的工作流网模型。

在图 6.19 中，起始库所中的托肯代表了一个客户的投诉，由此激活了整个工作流网。首先是“登记”活动被使能，它是一个由人工触发的活动，同时也是一个“与分支”，并行地将开始“寄出调查表”和“投诉评估”两个活动。“寄出调查表”是一个自动触发的活动，由邮件系统自动向投诉的客户发出电子邮件。“寄出调查表”活动之后，是一个“隐式或分支”，在“过期处理”与“调查表处理”这两个活动中二选一，它们分别是由时间触发和消息触发的，哪一个活动先被触发，则执行哪一个活动。执行完其中某一活动后，将有一个托肯被移至库所 c5 中。再来看“投诉评估”活动，它是一个“显式或分支”，由活动的处理结果来决定向 c6 或 c7 库所中转移托肯。如果向 c6 中转移托肯，则表示不处理投诉，直接执行归档活动；如果向 c7 中转移托肯，则表示要处理投诉，相应地，“处理投诉”活动将被执行。“处理投诉”既是一个“与连接”，又是一个“与分支”：“与连接”保证了对投诉的处理必须在评估活动与调查表处理（或客户超过期限）都已完成的情况下才能进行；“与分支”则把由 c5 转移来的托肯再传回 c5，以保证归档活动的使能条件。“处理投诉”执行完毕后，将执行“检验”活动，在这里用到了一个反复组件，使得“处理投诉”活动有可能被重复执行，直到符合要求为止。最后进入“归档保存”活动，这也是一个自动触发的活动，由相应的文档系统自动进行记录处理。当有托肯输出到终止库所中时，整个工作流网也就进入了结束的稳定状态。

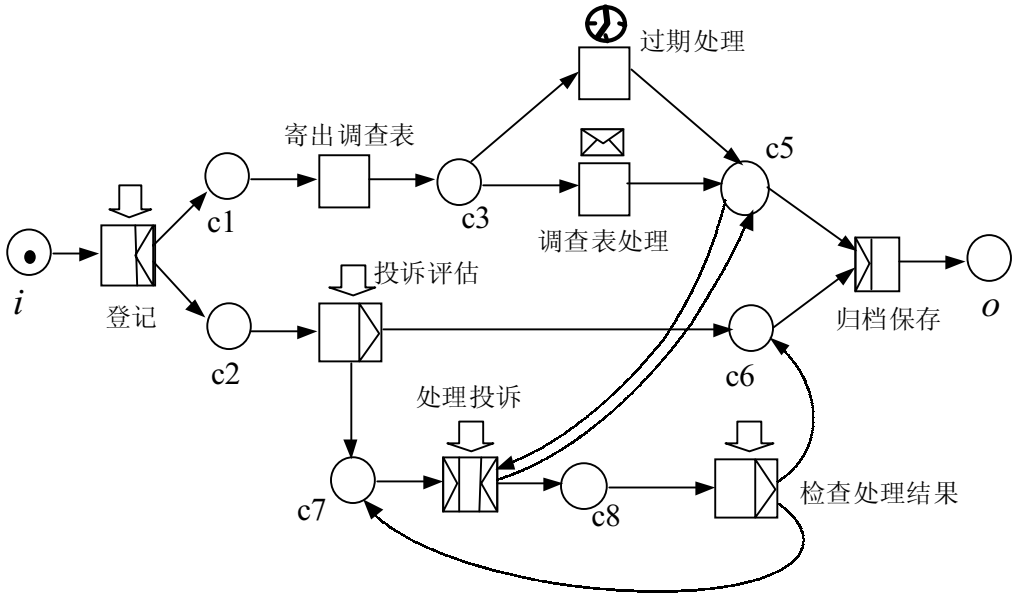


图 6.19 一个客户投诉的例子

对于 Petri 网而言，工作流管理是一个能够充分显示其强大实力的应用领域。通过工作流网的定义，利用不同的基本组件与触发机制，可以完成对复杂过程逻辑的设计，上面的一个例子就说明了这一点。对于上面的模型实例，许多工作流系统在模型定义方面是存在困难的，即使能够表示出这样的过程，在清晰度与准确性等要求上也无法与基于 Petri 网的工作流网模型相比。而且，Petri 网建立在严格的数学基础上，已经有了许多成熟的分析方法和工具，能够从理论与仿真两个方面对工作流网的多种性质进行验证与评价，比如 Aalst 就对工作流网的“完全性（Soundness）”进行了论证，分析了不同的网结构对“完全性”的影响，并建议了一些合理的组件结构。这对于即将投入运行的工作流过程而言是非常重要的的，对企业经营过程的优化与重组也是一种有力的支持手段。

但是，从另外一个角度来看，Petri 网之所以具备上述的这些优点，实际上是在模型的构成上通过增加模型组成元素来实现的。与其它类似的模型相比，比如活动网络图，Petri 网实际上是把过程的状态通过库所中的托肯予以显式地表达，而活动网络图则因为没有库所及托肯这样的元素定义只能隐式地或通过其他方法来表达相关状态。因此，这就给 Petri 网带来了一个必然的不良影响——组成模型的元素数量过多。一个活动网络图中的活动在 Petri 网中一般需要一个库所、一个变迁以及连接其间的一条连接弧来表示，这必然使模型变得复杂化。即使是一个比较简单的过程，其相应的 Petri 网模型也会有较多数量的库所与变迁；对于复杂的过程，这一问题则更显突出。对于非专业人员而言，理解 Petri 网流程要比理解活动网络图困难得多。目前市场上以 Petri 网为建模界面的工作流产品仅有少数的几

种（如 COSA、INCOME、LEU），在数量上远远不及其他模型界面的产品。

Petri 网的另一缺点是在 Petri 网中无法体现数据流。尽管基于状态建模的 Petri 网能够精确、方便地对过程的控制逻辑进行定义，然而在形式上一个 Petri 网也只能体现这些内容。在这种情况下，数据流就只能与控制流完全混合。当数据流逻辑与控制流逻辑不一致时，Petri 网就无法显式地表达这种独立于控制流之外数据流。

在基于 Petri 网的工作流网定义中，每个活动被映射为一个网中的变迁，对于这个变迁而言，在被触发后，它发生转移是需要一定时间的。在这样的一个时间段中，它还有可能经历一些不同的状态，比如正常执行、异常处理、活动挂起等。如果在工作流的过程建模中需要用到这些内部状态，比如活动 A 的执行不能早于活动 B 的执行（也就是说，在触发 A 活动时，B 活动应该已经处于执行状态），那么仅仅用一个变迁来表达活动是不能满足要求的，需要对变迁进行分解，即引入子网变迁的概念，以建立变迁的内部精细结构。

建立一个工作流网，就可以在上面运行多个工作流实例，通过对起始库所中的托肯加颜色或者给予不同的标识号，即可区分不同的工作流实例，并可同时对它们进行监控。如果在运行当中某一库所存有的托肯数量较多，则表明由该库所使能的活动对于整个过程的处理能力而言是一个瓶颈，需要对其进行改进，以增加过程的吞吐量。

6.6 工作流的事务模型

事务已经成为设计与编制高可靠性软件系统的最重要的概念。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。事务通常指一组对于物理的或者抽象的应用状态的操作组成的集合。事务处理系统通常指一个完整的系统，包括应用生成器、操作工具、数据库、外设、网络和操作系统。一个事务处理系统提供了工具来方便应用编程，并通过这些工具来提高应用执行和监控工作的自动化程度。事务与普通的程序操作的最大区别是它具有的事务特性，简称 ACID 特性。事务所具有的四个特性为：

（1）原子性（Atomicity）：所有关于一个事务的操作必须当成一个原子单元，即或者所有的操作都成功执行，或者一个也没有执行。如对数据库中数据所做的一系列操作必须全部完成后才能提交，否则所做的部分操作必须全部取消；

（2）一致性（Consistency）：事务操作得到的结果必须保证是一致的，如对于数据库的操作生成的新的数据库状态（数据）必须满足数据库的一致性（完整性）约束；

（3）分离性（Isolation）：在事务状态下执行的任何操作如同它在一个单用户环境下执行

操作那样，它不受其它并行的操作执行的任何影响；

（4）持久性（Durability）：事务提交后，对数据的操作结果不会丢失，将持久保留。

事务特性的提出主要是为了提高数据库的出错恢复能力和数据库系统的可靠性，目前事务处理的概念已经在许多应用领域得到了应用。经过研究人员多年的工作，在事务处理领域已经形成了许多用于并发控制、死锁预防、出错恢复等的一系列机制和方法，这些方法已经成为编制分布应用环境下具有高可靠性的大规模复杂系统重要基础。

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。比如，工作流中的某一活动实例正在对一用户数据进行写操作，此时就不应该再允许其他的活动实例同时对该数据进行有关的读写操作；还有，当一个工作流过程在执行到某一活动时被迫取消，那么，在这一活动之前已经完成的活动所造成的影响也就应该尽可能地取消（当然，有些操作结果是无法取消的），特别是被改动的数据，应该把它们恢复成过程执行前的状态。为处理这些情况，都需要在工作流管理系统中引入事务概念和相应的事务处理方法。但是，工作流的事务处理要比数据库中的事务处理复杂得多，因为工作流具有更复杂的操作、更广的分布性以及更多的异构性。

最初的工作流管理系统主要是为了实现局部与集中环境下办公流程与文档管理的自动化，因此，在今天的分布、异构环境与复杂的企业经营过程下，工作流管理系统面对所出现的并发操作与操作失败等情况缺乏保证执行正确性与可靠性的能力。目前的工作流产品和原型系统所具有的事务处理能力都远远不能满足实际企业的要求，这也是工作流管理系统在实际应用推广中所遇到的主要障碍之一。

为了更好地描述工作流的事务特性和解决工作流的事务处理问题，人们首先在数据库事务模型的基础上提出了许多高级事务模型（Advanced Transaction Model），包括嵌套事务模型、多层事务模型、Sagas、分支/汇合事务模型、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。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。ConTracts 模型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；Amit Sheth 在对这些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。下面我们对一些主要的高级事务模型以及 ConTracts 模型、事务工作流进行介绍。

6.6.1 嵌套事务模型

从经典事务模型到高级事务模型，一个很重要的改进步骤就是由单层结构扩展为多层结构。嵌套事务（Nested Transaction）模型由 Moss 提出，它是把一个事务分解为多个子事务（Sub-transactions）；同时，子事务也可以继续分解成为更细的子事务，由此形成了一棵“事务树”。如图 6.20 所示。

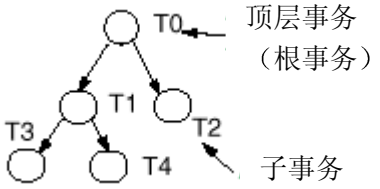


图 6.20 事务树的分解

子事务只有在其父事务开始后才可以开始，而父事务只有在所有的子事务全部结束后才能结束。如果父事务退出，则所有的子事务必须退出，包括已经提交的子事务也必须撤销；如果某个子事务失败，父事务可以选择自己的恢复方法，比如执行一个替换性操作，称为意外子事务（a contingency subtransaction）。同一层的子事务之间可以同时处于活动状态，但一个子事务不能看到另外一个可能退出的子事务的更新结果。嵌套事务模型在全局层次上保证了完全的分离性，同时增加了事务的模块化程度，在失败处理时具有更细的粒度，在事务内部具有更高的并发性，减少了处理的响应时间。

嵌套事务和软件工程中的模块化技术具有非常大的相似性。实际上，将嵌套事务模型的概念与软件工程的模块化方法结合起来，利用它们之间的互补性可以得到较理想的实施效果：模块化设计完成应用的结构化和局部数据结构的封装，嵌套事务模型保证这些模块使用的全局数据具有独立性并且是可恢复的。

嵌套事务模型在一些系统中已经得到了实现，包括卡内基梅隆大学的 Camelot 原型系统、加州大学洛杉矶分校（UCLA）的 Locus 系统、Transarc 的 Encina 事务处理系统、麻省理工的 Argus 项目等[44]。这些系统有的已在分布式环境下得到了应用，如基于 DCE 的 Encina 提供了事务性远程过程调用（Transactional RPC），把事务由一个应用延伸到了另一个应用，RPC 的失败就意味着事务的退出。

6.6.2 Sagas

Sagas（来自古挪威语，传奇，也有长篇故事的意思）是由普林斯顿大学的 H. Garcia-Molina[45]等人提出，用于解决长时间事务问题的一种模型。这个模型的基本思想是允许事务在全部提交之前能够释放某些占用的资源（数据），这一点对于持续时间很长的事务而言是很有意义的，它大大地提高了资源的利用率，同时也增强了系统的事务处理能力。在这里，一个长时间的事务被称为一个“Saga”，它由一个预先定义好执行顺序的子事务集合 T 和一个相对应的补偿子事务集合 CT 组成。子事务集合 T 是一个子事务序列 T_1、T_2、……、T_n，补偿子事务集合 CT 则包括了集合 T 中的每一个子事务所对应的补偿子事务 CT_1、CT_2、……、CT_n。任意一个子事务 T_k 都具有 ACID 特性，以保证数据库的一致性。一旦子事务 T_k 完成，通过释放锁定，它的操作结果对其他事务而言就是可见的。只有当 T_1、T_2、……、T_n 按顺序依次提交后，一个 Saga 才被成功地完成；如果其中的某个子事务执行失败，比如是第 k 个子事务 T_k，那么系统将通过执行补偿子事务来撤销 T_k 和前面已经提交的 k-1 个子事务的操作结果，补偿子事务一般都是逆序执行的，这样，整个过程的执行顺序为 T_1、T_2、……、T_k、CT_k、CT_{k-1}、……、CT_2、CT_1。由于各个子事务间的执行是完全串行的，因此这一模型又被称为线性 Sagas，是一种基本的 Sagas 模型。

Sagas 放松了对事务操作的分离性的要求，增加了事务间的并发性。在线性 Sagas 的基础上，H.Garcia-Molina 等人又进行了扩展，提出了并行 Sagas 模型和一般 Sagas 模型，引入了必要的并行机制和嵌套机制。

Sagas 为研究人员提出了一些有价值的想法：把一个需要长时间运行的事务分解成几个部分，同时在各个部分之间建立明确的控制流与数据流。这种思想与工作流的过程定义非常类似，只不过后者具有更为复杂的形式。当然，Sagas 也存在着一些未解决的问题，如其概念至今并没有得到实际系统的实现，另外还有分布式处理问题、重用问题、补偿事务的死锁问题等等。

6.6.3 分支/汇合事务模型

分支/汇合事务（Split/Join Transaction）模型是由美国哥伦比亚大学的 G. Kaiser 和 C. Pu 提出的，是针对“末端开放（open-ended）”的活动而专门设计的。所谓“末端开放”是指活动具有不可确定的执行时间、不可预知的发展方向、与其他并发的活动进行交互等特点。这种高级事务模型通过动态重构正在进行的事务，解决了多个用户间的合作问题。“分支”

命令可以把一个正在进行的事务分成两个事务，“汇合”命令可以把两个事务合并成为一个事务。与其他事务模型相比，这种模型在初始化的事务集合与最后提交的事务集合之间不存在简单的对应关系，因为在事务执行的过程中经历了分支/汇合的操作，使得事务被重新构造，可能无法与最初的事务相对应。

如图 6.21（a）所示，事务 T1 通过“分支”形成两个事务 T1a 和 T1b，那么由 T1 执行过的所有读/写操作将被分配给 T1a 和 T1b 这两个事务，T1a 和 T1b 将负责这些读/写操作的提交与退出。事务 T1 分支的前提是 T1a 与 T1b 是可串行化的，这将保证系统的正确性。根据产生冲突的可能性，“分支”可以有“串行化分支”和“独立性分支”两种：“串行化分支”所产生的两个事务在提交顺序上是有依赖关系的，比如 T1b 的提交必须在 T1a 提交之后；而“独立性分支”所产生的两个事务在提交与退出方面则相互独立，也就是说，T1a 与 T1b 可以任意提交或退出。

图 6.21　分支/汇合模型示例

在事务“汇合”时，如图 6.21（b）所示，T2 被合并到了事务 T1，那么类似地，T2 的读/写操作集合将被加入到 T1 当中。在“汇合”变为有效与 T2 消失之前，T1 必须接受 T2 的合并。在此之后，T1 就能够使用 T2 占有的所有资源，而且 T2 的操作结果也将在 T1 提交后变为持久有效。需要说明的一点是，不加控制的事务合并将导致系统的非串行化，比如，事务 T_A、T_B、T_C 具有 $T_A \prec T_B \prec T_C$ 的串行化顺序，那么 T_A 与 T_C 的合并将会出现错误。

分支/汇合事务模型实现了一种多用户的合作行为[47]，通过“分支”与“汇合”能够把部分事务的处理工作分配给某一合作者。另外，如果分支后的某一事务能够立刻提交，这将增加系统的并发处理能力，因为这将比原来的事务能够更早地释放某些资源并产生持久的操作结果。这种模型也存在一些很棘手的问题，包括如何能够保证分支后的两个事务能够再次形成有效的工作单元，如何定义资源的分配等。

6.6.4 ACTA

ACTA是由麻萨诸赛大学的Ramamritham等提出的一个综合性事务框架[48]，它具有足够的柔性用以实现对多种扩展事务模型属性的形式化描述与推理。ACTA在拉丁文中的意思是“actions”，指构成计算的多个动作（操作）。顾名思义，ACTA强调自身是一个描述这些动作（操作）属性的框架。

与前面提出的事务模型相比，ACTA并不是一个单独的事务模型，它实际上是一个元模型。在ACTA提出之前，包括嵌套事务模型、多层事务模型、分支/汇合事务模型以及Sagas等在内的多种高级事务模型已经出现。与经典事务模型相比，这些高级事务模型所要适应的应用更具复杂性与交互性，比如访问较多的对象、长时间计算、被用户暂停等等；同时这些模型分别在事务的A、C、I、D 四个方面放宽了要求，也就是说，为了追求较好的性能而修改了原有经典模型的属性。针对这些高级事务模型在经典模型的基础上所做出的不同的修改，Ramamritham等人提出了如下的问题：（1）一个事务模型在A、C、I、D 方面各自拥有怎样的属性？（2）如果这些属性能够被准确地定义出来，那么就有可能确定一个事务模型与另外一个事务模型之间的关系，比如二者的相似性与区别。（3）从实现的观点来看，需要怎样的机制来管理这些高级事务模型？特别是需要怎样的机制来实现想要得到的正确性属性？这些问题的提出也正是ACTA产生的背景。

ACTA把事务的属性表达为可见性（Visibility）、一致性（Consistency）、恢复性（Recovery）以及持久性（Permanence），每一个属性都有相应确定的依据。ACTA允许采用重要事件（Significant Event）来表达事务之间的相互依赖关系，这比单纯地用普遍存在的提交（Commit）和退出（Abort）事件来表达事务间的交互要更强大。这些重要事件包括开始（Begin）、准备（Prepare）、提交（Commit）、退出（Abort）、产生（Spawn）、分支（Split）、汇合（Join），它们被定义成原语，从而在事务之间建立了不同的依赖关系，包括提交依赖（Commit Dependency）、退出依赖（Abort Dependency）、开始依赖（Begin Dependency）等等。事务间的依赖关系将随着新的重要事件的定义而不断增加，从这一点来说，ACTA是一个具有开放性质的框架。ACTA并未对各类高级事务模型的结构做出任何假设，因此，ACTA不只限于表达层次化结构的事务。

ACTA采用一阶谓词逻辑的演算形式，比如给定一个对象的状态s，操作p在s的基础上所产生的输出被表示为 return（s, p）；p操作后对象所处的状态表示为 state（s, p）。ACTA的组成部分如图6.21所示，分两个方面：在事务方面，主要为事务间的依赖关系，如上文

所述，它们是通过重要事件来进行定义的，比如提交依赖被表示为（tj CD ti），即如果事务ti与tj提交，则ti必先于tj提交。在对象方面，主要包括事务视图、事务冲突集以及代理三个部分。事务视图定义了在某一时刻对事务可见的对象状态，事务冲突集则包括了正在进行中而且可能产生某种冲突的操作。代理主要用来定义事务间职责的移交，因为通常来讲，发起操作的事务将负责提交或者退出该事务，但在某些情况下，发起操作与提交或退出并非由同一个事务来完成，这就发生了职责的转移。代理不仅被用于控制对象的可见性，还可用于定义事务模型的可恢复属性，这些功能在具有合作性质的环境中是很有用处的。用户正是通过以上这些基本组件来定义整个事务模型的行为属性。图6.22给出了ACTA的框架结构。

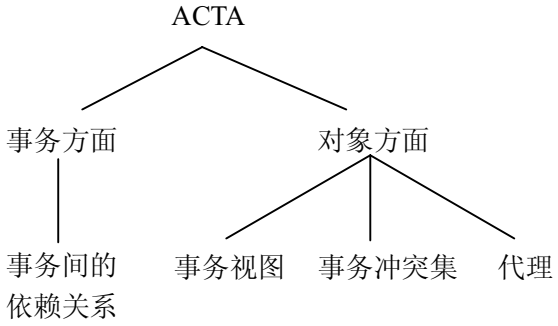


图6.22 ACTA框架

在ACTA框架下，一个事务模型的定义是由一个公理集所构成。在这个公理集中，通常定义了有关该模型下事务历史的不变式断言，或者是操作的前条件、后条件以及事务管理原语。ACTA支持对已经存在的各种事务模型的定义和分析，Ramamritham等人已把原子事务、分布事务、嵌套事务以及分支/汇合事务、Sagas用ACTA框架进行了描述，给出了每一种模型的公理化定义。除了能够对已有模型进行形式化定义以外，ACTA还可以通过对属性的修改和综合，来建立新的事务模型，这些新的事务模型可以由用户针对不同的应用环境与应用需求进行定制，所以，ACTA还是一种开发新型事务模型的有力工具。

有关 ACTA 的详细内容，可参考文献[48]。

6.6.5 ConTracts

ConTracts 模型[26]是 Andreas Reuter 等人在德国 Stuttgart 大学的 ConTracts 研究项目中所提出的。该项目也是为寻求解决数据库中长时间计算过程（事务）的方法而开展的。在项目开始时，人们已经针对这个问题提出了诸如 Sagas、柔性事务、分支事务等高级事务

模型，但是 ConTracts 项目组的研究人员认为：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，不具备完备的 ACID 特性的事务实际上是没有应用价值的；对长时间事务的管理需要一个系统的方法，不能仅仅在原有模型的基础上修修补补；对于一般简单的事务可能只在系统中存留 10^{-1} 秒，而对于可能执行几年的过程来说，它将存留 10^{8} 秒，这种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。

ConTracts 项目的研究范围不仅仅限于数据库，还拓展到了工作流管理领域，这是一个很大程度的进步，因为在此之前的相关研究并未真正把工作流做为研究对象。研究人员对几点比较关键的、与工作流有关的用户需求进行了如下分析：

（1）用户对工作流的历史进程非常关心，他们希望能够获得一些过程执行的信息，比如，工作流执行了哪一条路径，谁完成了什么任务等。

（2）在工作流系统中，前向恢复是必不可少的，因此，与之相关的信息也必须容易获得，而且与过程本身一样，这些信息也应该具有持久性和可恢复性。

（3）在定义阶段，用户需要高级环境的支持，比如图形或文字；没有人愿意直接书写底层的执行代码来定义过程。

（4）在工作流过程中，应该明确地定义步骤之间的信息流。

（5）对于工作流过程而言，完全的分离性（Isolation）既不需要，也没有意义。

（6）实际的应用需要一种柔性的机制来表达它们所要求的分离性。

（7）由分离性问题所导致的冲突不可能由自动系统机制来解决，必须支持应用自身的冲突解决定义。

（8）工作流过程的正确性规则不能是可串行化（Serializability），需要定义新的规则。

在这些需求的驱动下，ConTracts 在两个方面做出了很大的贡献：定义了一种高级数据模型；提出了一种高级并发控制机制。这也成为 ConTracts 模型的两个重要特色。下面我们对 ConTracts 模型做简要的介绍。

一个“ConTract”可以理解为一个过程，它是由一系列定义好的步骤（Step）按照一定的控制流脚本（Script）进行执行。如图 6.23 所示，给出了一个“商务旅行预约”的脚本。

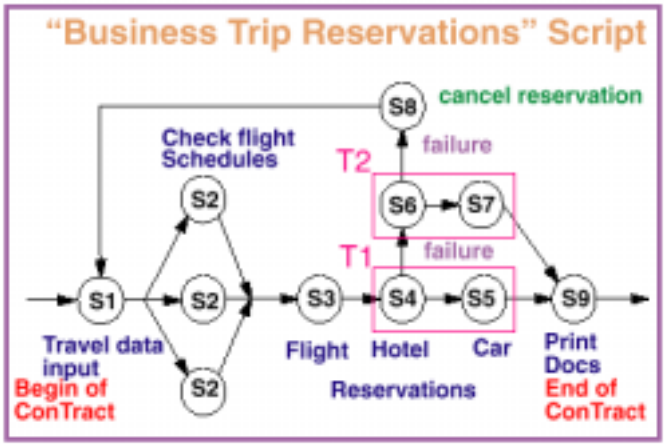


图 6.23 ConTract 的一个脚本

在 ConTract 脚本中，包含了对 ConTract 局部变量的声明，这些变量就是构成 ConTracts 数据模型——Context 的基本元素，它们被保存在可靠的、事务性的存储器中，只能由定义它们的 ConTract 来访问。因此，这些变量将记录 ConTract 中每一步骤的提交情况。为支持用户对工作流历史进程的访问，每一个 Context 元素都具有五维模式，即一个五元组，包括变量名称、ConTract-ID 号、Step-ID 号、循环计数、并行索引。这种五维的模式要比通常基于时间版本的二维模式能够更有效地支持历史的记录和检验。

为了避免全局的分离性，那么过程中的某些操作结果有可能在过程尚在执行的过程中就已对外可见了，因此，ConTracts 模型引入了补偿机制来撤销原来执行步骤的结果。一个 ConTract 是正确的，当且仅当它处于初始状态或者处于结束状态或者每一个步骤的补偿步骤都能够保证执行。为了使补偿步骤能够执行，ConTract 为步骤建立了约束条件，这些约束是以谓词逻辑的形式表达的。在执行阶段，当一个步骤执行完毕后，相应的谓词就被检验：若有效，则保护该步骤的事务就允许提交；若无效，则通知 ConTract 管理器，ConTract 管理器将获取冲突的有关信息，以便能够初始化补偿步骤来解决冲突。这种机制在保证系统正确性的同时，还尽可能地解除了不必要的访问限制，从而提高了系统的性能。

基于 ConTracts 模型，该项目还开发了一个模型实现的原型系统，称为 APRICOTS。系统是用 C 语言和 SQL 实现的，数据库为 INGRES。系统中的编译器把脚本代码转换成为谓词转移网络（Predicate Transition Network）；日志和编译后的脚本都存放在数据库表中。系统的体系结构和 ConTract 管理器的内部结构如图 6.24 所示。

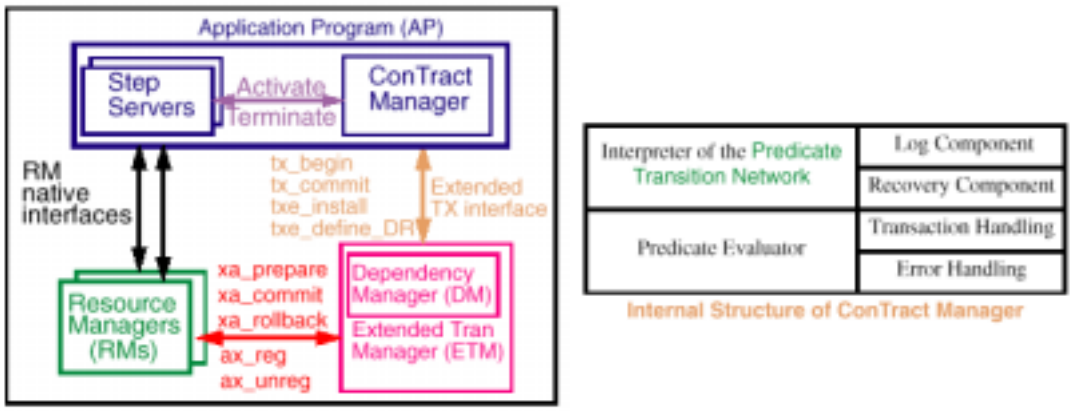


图 6.24 APRICOTS 的体系结构与 ConTract 管理器的结构

ConTracts 模型也同样有一些未解决的问题，比如，无法处理已经存在的一个步骤中包含多个事务的应用，对嵌套 ConTracts 的支持，对动态模式修改的支持，补偿事务的运行次序问题等等。但总的来说，ConTracts 模型脱离了其他高级事务模型研究方法所普遍存在的局限，并把目标直接拓展到工作流领域，这是一个很有意义的进步。

6.6.6 事务工作流

事务工作流是由 Georgia 大学 Amit Sheth[28]等人最先提出的，这一概念强调了与工作流密切相关的事务属性。一个事务工作流包含了多个任务的协作运行，这些任务可能要访问到异构的、自治的、分布的数据库系统。任务间的协调是通过基于相互依赖的控制流方法进行描述的，它为每一个任务定义了执行的先决条件，这些条件可以是基于其他任务的执行状态（比如某一任务是否提交、是否退出或结束等），也可以基于其他任务的输出参数，或者基于某些外部变量（比如时间）。

前面提到的多种高级事务模型可以被用来建立工作流的模型，其中的每一个子事务就对应了工作流中的任务，而事务的执行结构就对应了工作流中的控制流。这样，工作流的分离性与原子性就完全由它所使用的高级事务模型所决定。Amit Sheth 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（因为工作流系统不能保证所有的任务处理实体都是具有事务管理特征的 DBMS）；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。因此，Amit Sheth 提出了对事务工作流的一些有意义的处理方法，主要包括任务的定义、依赖与正确性规则、工作流的执行等。我们在这里只对任务定义和依赖

规则进行简要介绍，详细内容见文献[27][49]。

（一）任务的定义

工作流中的任务是一个工作单元，它可能是传递一个消息，也可能是填写表格，或者是执行一个过程等等。对任务的一个抽象模型就是状态机，通过状态转换图即给出了任务的一个轮廓。在工作流的层次上，不需要对任务的内部操作进行定义，只需要处理那些对外可见或者需要加以控制的部分。通常情况下，每个任务可能因具有不同的内部结构而表现为不同的外部轮廓。如图 6.25 所示的例子，分别给出了非事务性任务、事务性任务以及两阶段提交的事务性任务这三类任务的状态转换图。

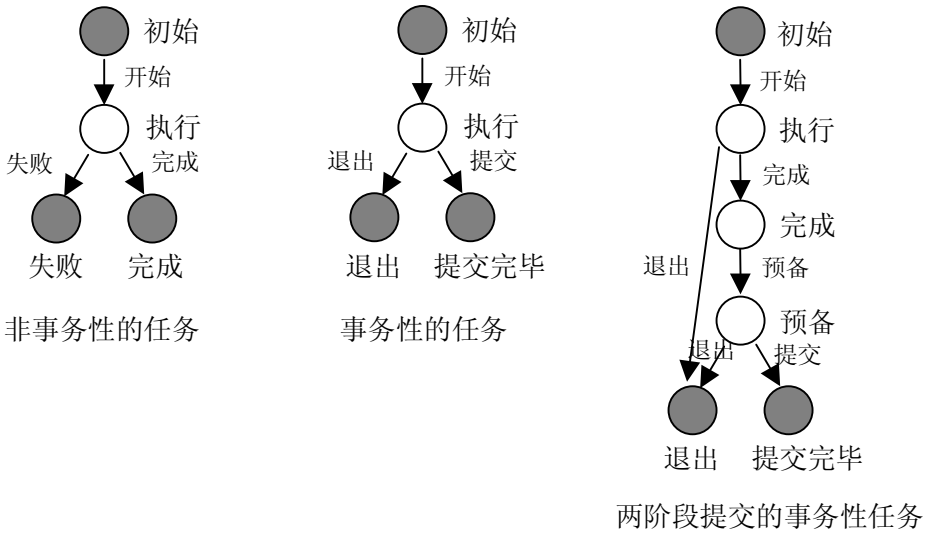


图 6.25 几种不同类型任务的状态图

对于一个任务的定义，应该包括：（1）一个对外可见的任务执行状态集合，在这个集合中应该至少有一个初始状态和一个或者多个终止状态。上图中的圆圈即表示状态，圆圈旁边的大号字体则标出了状态的名称，其中涂成深色的圆圈表示初始态和终止态；（2）一个导致状态转移的事件集合，当这些事件发生时，任务就由一个内部状态转移到另外一个状态，如上图中用小号字体标出的开始、失败、退出等。

在定义任务时可以不考虑执行它的实体，也可以考虑执行实体的能力与行为。对于前一种情况，确定哪个实体能够执行该任务或者由工作流系统模拟出执行实体所不支持的状态是很必要的；对于后一种情况，任务将被定义由特定的实体来执行，这包括一些已经存在的应用系统，这样，任务的外部轮廓将在很大程度上受执行实体的特点所决定。

当任务是由一个能够提供全面事务管理功能的数据库管理系统所执行时，我们就可以充分利用这些本地的并发控制、恢复机制来保证数据的一致性；但是当任务是由一个非数据库的应用系统执行时，我们就需要考虑可能影响事务特性的语义，这比在全局范围内建

立一种新机制来保证全面的事务性要更为合理有效。

（二）依赖规则

当确定了工作流中的所有任务后，就可以通过建立任务间的相互依赖关系来确定工作流的内部结构。这些依赖关系既可以是在执行前确定的（静态），也可以在执行过程中动态确定，这些依赖关系就构成了工作流中的控制流。如图 6.26 所示的例子，它描述了一个复合任务 A 的内部结构，任务间的箭头就表示基于状态的依赖关系。

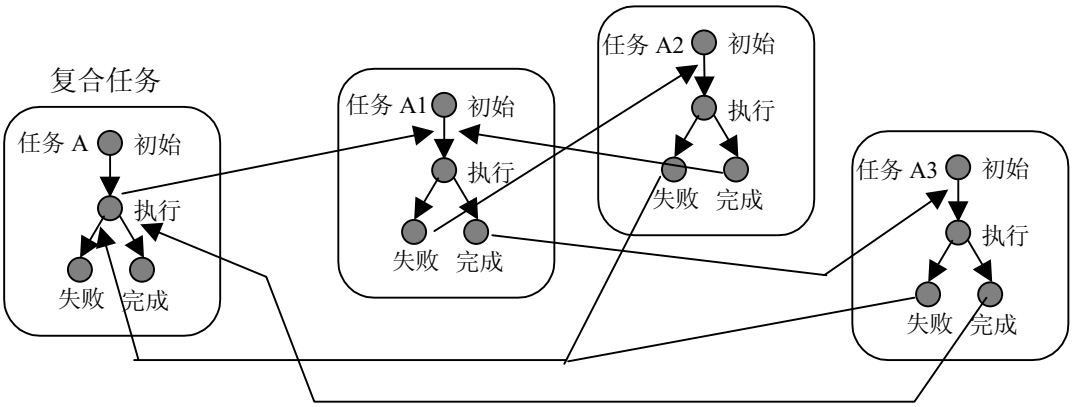


图 6.26 一个表示依赖关系的例子

在工作流的任务之间还可以通过变量来进行联系，这些变量可以为一些程序的执行携带参数，从而构成某一任务的输入/输出数据，即工作流的数据流。

在本章的最后，我们对工作流模型与事务模型的一些特性进行对比，以便找出二者的不一致之处，有利于更好地解决工作流的事务问题。从本章对各种模型的论述来看，工作流所考虑的四类实体为：用户、活动、程序和数据，工作流管理系统则实现了活动间控制流与数据流的自动化，并且将活动映射给用户与程序；而现有的事务模型则并未涵盖如此多的内容，往往只限于解决其中的一部分问题，具体对比起来，有如下几个方面：

（1）工作流模型具有描述组织的能力，并使工作流过程的定义与执行能够适应该组织的特点；而事务模型则完全没有这方面的概念；

（2）工作流中的活动并不仅仅限于计算机应用程序，它是为一般性的企业活动而设计的，而并非专门针对事务操作而设计，只要该活动能够向工作流管理系统通知其进展情况就可以被容纳于工作流系统之中；相反，事务模型则有很强的针对性，主要是在数据方面的操作；

（3）工作流管理系统的用户使用工作表与系统交互，系统采用一定的机制实现任务表在不同用户间的分配，以决定一个任务最终将被哪一个用户执行。分配的机制有多种，如 PULL 型、PUSH 型、随机型等等。相对而言，对工作表的处理功能也是事务模型所

不具备的；

（4）工作流模型目前一般不具备恢复与失败处理的语义，通常是依靠管理人员的手工参与来实现此功能；而事务模型则在自身的定义中重点对这方面内容进行了约束，提出了一些相应的解决方法，不同的约束就构成了不同的事务模型。从这一点来看，工作流模型正好需要引入一定的事务处理能力，一方面可以提高工作流系统的性能，同时也是对事务模型中相应解决方法的一个验证。

第 7 章　工作流过程定义语言 WPDL

在工作流管理联盟所提出的工作流系统参考模型中，以工作流服务为核心共定义了五类接口，从接口 1 到接口 5。有关过程定义的引入与导出构成了接口 1 的主要功能，工作流过程定义语言 WPDL（Workflow Process Definition Language）就属于这部分内容。

在实际应用中，人们会使用各种各样不同的工具来对经营过程（工作流）进行建模、归档和分析。工作流过程定义接口（接口 1）定义了一个公共的交换格式，使得不同产品的工作流定义可以实现模型交换。接口 1 还实现了工作流的定义与执行两个不同的阶段的分离，使得工作流的定义工具与执行工具相互独立，从某一个定义工具中建立的工作流模型可以作为多个不同工作流机（或运行时系统）的输入。这一特性大大增强了用户对工作流管理系统的选择范围与可配置性，他们可以依据自身业务的特点选择符合工作流管理联盟标准的、来自于不同厂商的产品模块共同组成一个完整的工作流管理系统。

为实现这一接口特性，工作流管理联盟定义了一个工作流过程的元模型，元模型中给出了一般的模型实体及相应的属性。WPDL 就是基于这一元模型而定义的文本描述语言。作为一个标准、通用的工作流定义语言，WPDL 定义了一个最小集合的工作流建模实体与属性，提供了一般意义下的公共交换格式。基于这一模型，特定厂商的工具之间可以方便地交换信息，如图 7.1 所示。

本章将介绍 WPDL 的标准语法格式，包括基本数据、属性以及各种工作流模型实体的定义方法，并给出一个用 WPDL 描述实际经营过程的例子，以加深读者对于这种基于文本形式的工作流过程定义语言的理解。

7.1 WPDL 语法及语言结构

WPDL 的语法采用类似于巴科斯－诺尔范式（BNF = Backus Naur Form）的格式，以一个产生式规则集合的形式进行定义的。语法（元语言）的组成包括以下几个部分：

标识符　　<example_symbol>

关键字　　EXAMPLE_KEYWORD

产生式标记　: : =

特殊字符　　[　]　|　/　*

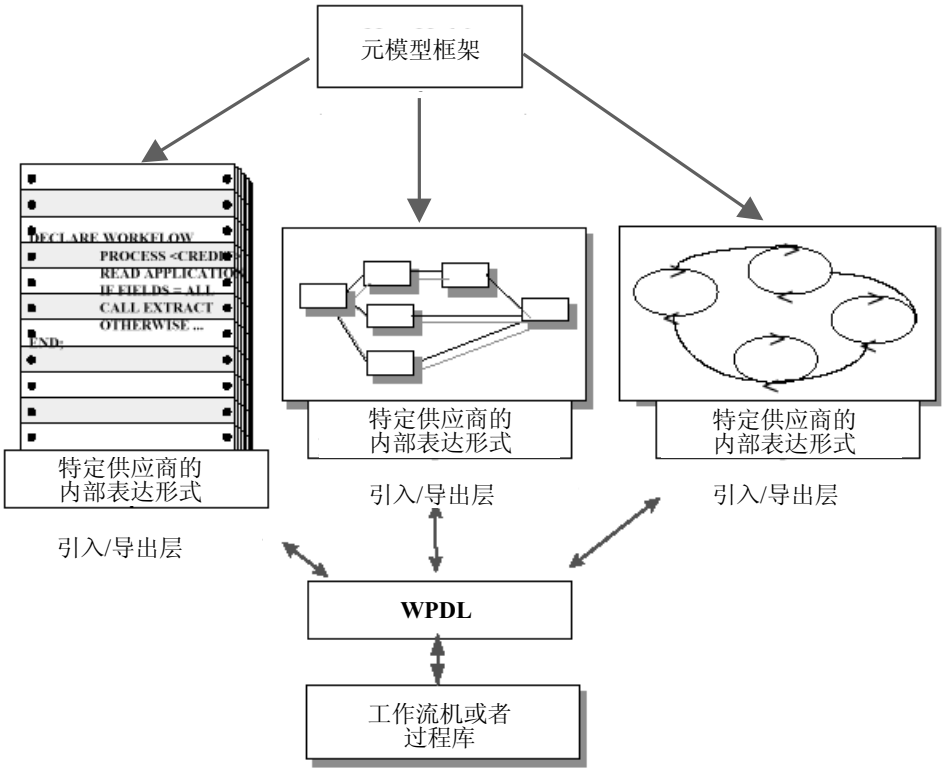


图 7.1 不同类型过程定义的交换

在 WPDL 中，每一条语法规则都是一个产生式，规则的左边（标识符）和右边（表达式）用“: : =”进行连接，其意思是左边可以由右边代替或由右边产生。右边的表达式可以是标识符、关键字以及特殊字符的组合。关键字是语言的核心部分，用来刻画 WPDL 过程模型中的各种组成部分以及相应属性，由空格符分隔，并且是区分大小写的；出现在方括号“[ ]”中的关键字或标识符表示是可选的，可以使用一次或者不用；特殊字符“ | ”的含义是“或者”、“二者选一”；特殊字符组合“ /* ”、“ */ ”以及“ // ”用来作为注释的标注，其用法与 C++编程语言中的注释用法完全相同：位于“ /* ”和“ */ ”之间的文字以及位于“ // ”至行末的文字都属于注释，而非有效语言内容。

以下是 WPDL 语法中对一些特殊字符的约定：

操作符以字母“Op”结尾，比如<NotOp>表示“非”操作。

常量以大写字母“C”结尾，比如<BooleanC>表示布尔型常量。

括号以大写字母“B”结尾，比如<OpenArrayB>表示数组定义中的一个括号。

类型字符用大写字母书写，而且以大写字母“T”结尾，比如<INTEGER-T>表示整数类型。

其他结束符号均以大写字母书写，比如<UPTO>。

7.1.1 基本数据类型、表达式以及操作符

在 WPDL 语法中，有些符号是最基本的，是用来定义其他符号的，这主要是指基本的数据类型、表达式以及操作符的定义。WPDL 对它们分别定义如下：

基本数据

<basic data> : : = <string> | <float> | <integer> | <reference> | <date>

其中，

<string> : : = 字符串，被引在双引号 ” ” 之间，比如”example”。字符串的最大长度为 1024，其中包括空字符 null 和引号在内。

<float> : : = 浮点型。

<integer> : : = （有符号）整型。

<reference> : : = 外部对象的指针，以字符串表示，比如文件名”c:\test\test.exe”、邮件地址”abc@def.ghi”、URL 地址”http://www.abc.com”。

<date> : : = 日期型，格式为 YYYY-MM-DD [hh:mm[:ss]]，符合 ISO 8601/EN 28601 标准。

间隔符

<blank> : : = WPDL 字符之间的标准分隔符，通常为空格符，也可以是回车符（CR）与/或换行符（LF）。

标识符

<identifier> : : = 由字母、数字、下划线、点及间隔符组成的序列，必须以字母或者下划线开头，被引在一对单引号之间。

基数

<cardinal> : : = 非负整型数，用于复杂数据结构中的基本符号。

简单数据

<plain data> : : = <basic data> | <boolean> | <performer>

<plain data type> : : = <basic data type> | <boolean type> | <performer type>

<basic data type> : : = <STRING-T> | <FLOAT-T> | <INTEGER-T>

| <REFERENCE-T> | <DATE-T>

<boolean type> : : = <BOOLEAN-T>

<performer type> : : = <PERFORMER-T>

其中，basic data 为前面的基本数据，boolean 为布尔类型的数据，分别对应 TRUE（1）和 FALSE（0），performer 则是工作流参与者类型的一个实例。

复杂数据

<complex data type> : : = <plain data type>

| <RECORD> <Member_list> <END> // 结构类型

| <ARRAY>

<OpenArrayB> <cardinal> <UPTO> <cardinal> <CloseArrayB>

<OF> <complex data type> // 数组类型

| <ENUM> <element list> <END> // 枚举类型

| <LIST> <OF> <element type> // 表类型

// 具有动态的元素个数

<Member_list> : : = <Member> [<Member_list>]

<Member> : : = <complex data type> <Midlist>

<Midlist> : : = <Member id> [<Midlist>]

<Member id> : : = <identifier>

<element list> : : = <element> [<element list>]

<element> : : = <complex initial>

<element type> : : = <complex data type>

<complex initial> : : = <plain initial> | <OpenArrayB> <element list> <CloseArrayB>

| <OpenB> [<element list>] <CloseB>

<plain initial> : : = <simple constant> | <PerformerC>

<simple constant> : : = <basic data> | <boolean constant>

<boolean constant> : : = <BooleanC>

复杂数据主要是对数组、结构、枚举以及表的定义，这些数据通常用来表达工作流相关数据与有关的扩展属性。

表达式

<expression> : : = <RelExpression> [<BooleanOp> <expression>]

<BooleanOp> : : = <ANDOp> | <OROp>

<RelExpression> : : = <ArExpression> [<RelationalOp> <RelExpression>]

<ArExpression> : : = <Unary> [<ArithmeticOp> <ArExpression>]

<Unary> : : = [<NotOp>] <Primary> | <UMinusOp> <Primary>

<Primary> : : = <VarReference> | <OpenB> <expression> <CloseB> | <PrimaryConstant>

| <function access> | PARTICIPANT <participant id>

<PrimaryConstant> : : = <simple constant>

<VarReference> : : = <data id> [<VarQualifier list>]

<VarQualifier list> : : = <VarQualifier> [<VarQualifier list>]

<VarQualifier> : : = <OpenArrayB> <ArExpression> <CloseArrayB>

| <PERIOD> <identifier>

<function access> : : = <function id> <OpenB> <CloseB>

| <function id> <parameter map list>

<ActualParameterlist> : : = <ActualParameters> [<COMMA> <ActualParameterlist>]

<ActualParameters> : : = <expression>

<condition> : : = <expression>

<integer expression> : : = <expression>

<string expression> : : = <expression>

<performer expression> : : = <expression>

表达式主要用于工作流中各种类型条件（循环条件、转移条件）的定义，它是一个由操作符与操作数所组成的序列。所有的操作符都是左结合的，而且具有事先固定的优先级及使用规则，对操作符的具体定义如下。

操作符

<ANDOp> : : = AND

<OROp> : : = OR

<NotOp> : : = NOT | ! // 同一语义的两种不同表达形式

<RelationalOp> : : = = | != | < | <= | > | >=

<ArithmeticOp> : : = + | - | * | /

<UMinusOp> : : = -

操作符的优先级由低到高依次为：（OR）（AND）（= != < > <= >=）（- +）（* /）（*unary*: NOT ! -）。括号中的运算符属于相同优先级，按照左结合的约定由左向右进行计算。

其他符号

<BooleanC> : : = TRUE | FALSE

<PerformerC> : : = UNKNOWN

<OpenB> : : = (

<CloseB> : : =)

<OpenArrayB> : : = [// 为了与语法定义中的元符号' [' 有所区别

<CloseArrayB> : : =] // 为了与语法定义中的元符号'] ' 有所区别

<STRING-T> : : = STRING

<FLOAT-T> : : = FLOAT

<INTEGER-T> : : = INTEGER

<BOOLEAN-T> : : = BOOLEAN

<REFERENCE-T> : : = REFERENCE

<DATE-T> : : = DATE

<PERFORMER-T> : : = PERFORMER

<END> : : = END

<ARRAY> : : = ARRAY

<RECORD> : : = RECORD

<ENUM> : : = ENUM

<LIST> : : = LIST

<UPTO> : : = . . .

<OF> : : = OF

<PERIOD> : : = . // 主要用于在 RECORD 结构中选择一个成员变量

<COMMA> : : = ,

<COLON> : : = :

7.1.2 属性、扩展属性以及参数

用 WPDL 定义工作流模型，实际上就是对一系列元模型实体进行描述。每一个元模型实体都以相应的关键字作为开始，接下来依次为标识符、属性表、实体描述的结束关键字。

每个属性的定义格式如下：

<attribute> : : = <attribute keyword> <attribute description>

这里的属性既可以是预定义的标准属性，也可以是扩展属性。扩展属性主要是为方便每个独立的软件供应商去定义自己需要的属性集，通过这种方法，有利于在较短时间内使得 WPDL 能够为工作流产品的软件供应商所承认并接受。扩展属性的格式定义如下：

<extended attribute list> : : = EXTENDED_ATTRIBUTE

<attribute id>

<attribute type>

<attribute value>

[<description>]

[<extended attribute list>]

<attribute id> : : = <identifier>

<attribute type> : : = <complex data type>

<attribute value> : : = <initial> | <function access>

<description> : : = <string>

参数在工作流过程定义、工作流应用定义中是必不可少的，它负责在不同的过程、子过程以及应用之间进行数据的传递，一般形式参数定义的格式如下：

<formal parameters> : : = [IN_PARAMETERS <parameter list>] // 输入参数

[OUT_PARAMETERS <parameter list>] // 输出参数

<parameter list> : : = <parameter> [<parameter list>]

<parameter> : : = <data id> // 工作流相关数据

在实际调用时，实际参数与形式参数的映射有如下定义：

<parameter map list> : : = <OpenB> <parameter map> <CloseB>

<parameter map> : : = <ActualParameterlist>

<ActualParameterlist>把实际参数按照顺序依次映射给形式参数，也就是说，第一个实际参数映射第一个形式参数，第二个实际参数映射第二个形式参数，依此类推。正常情况下，实际参数与形式参数的个数与类型是一一对应、相互匹配的，否则，就必须提供相应的处理机制，比如将丢失的参数置为零值、忽略多余的实际参数等等。不同的软件供应商有着不同的处理方法，此处不再赘述。

另外，参数也可以是一个表达式。在这种情况下，表达式的值将先被计算出来并放入

工作流机的缓冲区中，缓冲区的内容将被用于实际参数与形式参数的映射。

7.1.3 工作流模型

前面介绍的 WPDL 中的有关数据类型、表达式以及参数等内容，从本质上来说并未体现工作流的内容，因为一般编程语言的语法也都有类似的定义。从本节开始，我们将介绍 WPDL 中的核心部分——元模型实体的定义。正是这些元模型实体才组成了一个完整的工作流模型。

在一个工作流模型当中可以定义多个过程，它们共享相同的工具与参与者，但是，工作流管理联盟则希望用户为每一个经营过程建立一个包含所有必要过程、相关工具以及工作流参与者在内的工作流模型。工作流模型的定义格式如下，它也给出了整个模型的定义结构。

<Workflow Model> : : = MODEL <model id>

<Workflow Model Definition Header> // 工作流模型头定义

[<conformance class declaration>] // 一致类声明

[<extended library declaration>] // 扩展库声明

[<external model declaration>] // 外部模型声明

[<Workflow Participant Specification>] // 工作流参与者定义

[<Workflow Application List>] // 工作流应用表

[<Workflow Relevant Data List>] // 工作流相关数据表

[<Workflow Process Definition>] // 工作流过程定义

END_MODEL

<model id> : : = <identifier>

在工作流模型的头定义中包含了该模型一般的说明性信息，比如所用 WPDL 的版本号、模型提供商的 id 号等等，完整的定义格式如下：

<Workflow Model Definition Header> : : =

WPDL_VERSION <wpdl version>

VENDOR <source vendor id>

CREATED <creation date>

[NAME <name>]

[DESCRIPTION <description>]

[<redefinable header>]

[DOCUMENTATION <documentation>]

[PRIORITY_UNIT <unit>]

[COST_UNIT <unit>]

[<extended attribute list>]

<name> : : = <string>

<wpdl version> : : = <string>

<source vendor id> : : = <string>

<creation date> : : = <date>

<version> : : = <string>

<unit> : : = <string>

在工作流模型的头定义中，有一部分是可重定义的头信息（redefinable header），它不仅可以在工作流模型的头定义中出现，而且还可出现在模型中所包含的任何一个过程的头定义中，其格式如下：

<redefinable header> : : =

[AUTHOR <author>]

[VERSION <version>]

[CHARACTERSET <characterset>]

[CODEPAGE <codepage>]

[COUNTRY_KEY <country key>]

[RESPONSIBLE <responsible>]

[STATUS <publication status>]

<author> : : = <string>

<characterset> : : = <string>

<codepage> : : = <string>

<country key> : : = <string>

<responsible> : : = <participant assignment> // 通常是一个组织单元或者人

<publication status> : : = UNDER_REVISION | RELEASED | UNDER_TEST

一致类声明主要用来描述模型中的定义所符合的一致性类型。一致类是工作流管理联

盟对所有工作流软件供应商的软件进行鉴定、验收以便确认其是否符合 WfMC 标准的一种等级分类。由于各家的工作流软件所基于的工作流模型并不一定与工作流管理联盟所定义的元模型完全一致，因此，工作流管理联盟通过设立不同的一致性类型来对不同的软件进行标定。WPDL 所定义的一致类声明如下：

<conformance class declaration> : : = CONFORMANCE_CLASS <conformance class list>

<conformance class list> : : = <graph class conformance list>

<graph class conformance list> : : = FULL-BLOCKED // 全限制

| LOOP-BLOCKED // 循环限制

| NON-BLOCKED // 无限制

NON-BLOCKED 对所定义的模型结构无任何限制，包括自环、分支的使用；LOOP-BLOCKED 则不允许在模型中出现自环，对于需要循环的活动，只能通过多次执行来实现；FULL-BLOCKED 则有更严格的限制，除了 LOOP-BLOCKED 对非自环的要求以外，还要求每一个分支（SPLIT）或者汇合（JOIN）点都要与相应的汇合或者分支点相对应，从而组成一个类似于 BEGIN-END 的模块。另外，对于“与分支（AND-SPLIT）”不允许设置任何条件，对于“异或分支（XOR-SPLIT）”则不允许有任何未定义的转移结果出现，也就是说必须要保证有一个后继活动被激活。

在工作流模型中还定义了执行过程所需要的扩展库，扩展库中的函数与过程将直接与工作流机绑定并在必要的时候进行调用或者激活，无需通过工作流管理联盟所定义的第二类接口来执行。扩展库中的元素定义如下：

<extended library declaration> : : = LIBRARY

<library element list>

END_LIBRARY

<library element list> : : = <library function> [<library element list>]

| <library procedure> [<library element list>]

<library procedure> : : = PROCEDURE <procedure id>

[NAME <name>]

[DESCRIPTION <description>]

[<formal parameters>]

[<extended attribute list>]

END_PROCEDURE

<library function> : : = FUNCTION <function id>
RESULT <result type>
[NAME <name>]
[DESCRIPTION <description>]
[<formal parameters>]
[<extended attribute list>]
END_FUNCTION

<procedure id> : : = <identifier>

<function id> : : = <identifier>

<result type> : : = <plain data type>

在 WPDL 所定义的工作流模型中，还允许引用其他的工作流模型或者在另外已提供好接口的工作流管理系统中所建立的模型，这些都是通过外部模型声明来进行定义的。

<external model declaration> : : = EXTERNAL_MODEL_REFERENCE
<external model reference>
[<extended attribute list>]
END_EXTERNAL_MODEL_REFERENCE
[<external model declaration>]

<external model reference> : : = <logical model reference>
| <physical model reference>

<logical model reference> : : = WM <model id>
[<Access restriction part>]

<physical model reference> : : = OM <reference>

7.1.4 工作流过程定义

工作流过程定义构成了工作流模型的主体部分，因为它包含了组成了模型的所有活动以及转移信息。完整的定义格式如下：

<Workflow Process Definition> : : =
WORKFLOW <process id>
<Workflow Process Definition Header>

[<extended library declaration>]

[<formal parameters>]

[<Access restriction part>]

<Activity List>

<Transition Information List>

[<Workflow Participant Specification>]

[<Workflow Application List>]

[<Workflow Relevant Data List>]

END_WORKFLOW

[<Workflow Process Definition>]

<process id> : : = <identifier>

<Workflow Process Definition Header> : : =

[CREATED	<creation date>]
[NAME	<name>]
[DESCRIPTION	<description>]
[<redefinable header>]	
[DURATION_UNIT	<duration_tag>]
[PRIORITY	<priority>]
[LIMIT	<duration>]
[VALID_FROM	<date>]
[VALID_TO	<date>]
[CLASSIFICATION	<classification>]
[<time estimation>]	
[DOCUMENTATION	<documentation>]
[ICON	<icon identifier>]
[<extended attribute list>]	

<process name> : : = <string>

<priority> : : = <integer>

<classification> : : = <string>

<documentation> : : = <string>

<icon identifier> : : = <string>

<time estimation> : : = [WAITING_TIME <duration>]
[WORKING_TIME <duration>]
[DURATION <duration>]

<duration> : : = <integer><duration_tag>

<duration_tag> : : = Y | M | D | h | m | s

7.1.5 工作流活动

在上节的工作流过程定义中，有一部分内容就是描述过程中所包含的全部活动的信息，即活动列表 <Activity List>。

<Activity List> : : = ACTIVITY <activity id>
[NAME <name>]
[DESCRIPTION <description>]
<Activity Kind Information>
[<Access Restriction part>] // 对数据的访问限制定义
[<Transition Restriction part>] // 转移限制定义
[<extended attribute list>]
END_ACTIVITY
[<Activity List>]

<activity id> : : = <identifier>

对于活动列表中的每一个活动，<Activity Kind Information>描述了该活动是怎样被执行的。

<Activity Kind Information> : : = ROUTE // 仅标识路由信息，没有相应的执行部分
| IMPLEMENTATION <implementation>
[PERFORMER <participant assignment>]
[START_MODE <mode>]
[FINISH_MODE <mode>]
[PRIORITY <priority>]
<simulation information>

[ICON <icon identifier>]

[DOCUMENTATION <documentation>]

<implementation> : : = NO // 不为工作流所支持

| APPLICATIONS <generic tool list> // 应用

| WORKFLOW <subflow reference> // 子过程

| LOOP <loop kind> // 反复

CONDITION <loop condition>

<generic tool list> : : = <tool invocation>

| TOOL_LIST <tool list>

[DESCRIPTION <description>]

[<extended attribute list>]

END_TOOL_LIST

<tool invocation> : : = <generic tool>

[<parameter map list>]

<generic tool> : : = [TOOL] <generic tool id>

| PROCEDURE <procedure id>

<tool list> : : = <tool invocation>

[<tool list>]

<subflow reference> : : = <execution> <process id> [<parameter map list>]

<execution> : : = ASYNCHR | SYNCHR

<loop kind> : : = WHILE | REPEAT_UNTIL

<loop condition> : : = <condition>

<participant assignment> : : = <performer expression>

<mode> : : = AUTOMATIC | MANUAL

<simulation information> : : = [INSTANTIATION <instantiation>]

[<time estimation>]

[COST <cost estimation>]

<instantiation> : : = ONCE [| MULTIPLE]

<cost estimation> : : = <string>

转移限制定义部分主要是对活动执行完毕后如何发生转移进行描述，也就是定义了与

活动相连的转移结构，比如分支、汇合等。

<Transition Restriction part> : : =

[<Inline Block Information>]

[JOIN <JOIN characterisation>]

[SPLIT <SPLIT characterisation>]

<Inline Block Information> : : = <begin block>

| <end block>

<begin block> : : = INLINE_BLOCK_BEGIN <block id>

[NAME <name>]

[DESCRIPTION <description>]

[ICON <icon identifier>]

[DOCUMENTATION <documentation>]

[<extended attribute list>]

END_INLINE_BLOCK_BEGIN

<end block> : : = INLINE_BLOCK_END <block id>

<block id> : : = <identifier>

<JOIN characterisation> : : = AND | XOR // 与连接、异或连接

<SPLIT characterisation> : : = AND | XOR <list of Transitions> // 与分支、异或分支

<list of Transitions> : : = <transition id> [<list of Transitions>]

7.1.6 转移信息

转移信息定义了过程中连接所有活动的转移弧的信息以及弧上所定义的转移条件，它们体现了过程的控制逻辑。

<Transition Information List> : : =

TRANSITION <transition id>

[NAME <name>]

[DESCRIPTION <description>]

<transition kind description>

[<extended attribute list>]

END_TRANSITION

[<Transition Information List>]

<transition id> : : = <identifier>

<transition kind description> : : =

FROM <activity id> TO <activity id>

[CONDITION <transition condition>]

| FROM LOOP <activity id> TO <activity id>

| FROM <activity id> TO LOOP <activity id>

<transition condition> : : = <condition> | OTHERWISE

7.1.7 工作流应用定义

工作流应用定义是一个应用及工具的列表，它们将在工作流的执行过程当中被激活或调用。在定义中当然也包括对激活应用所需参数的说明，具体格式如下：

<Workflow Application List> : : =

APPLICATION <generic tool id>

[NAME <name>]

[DESCRIPTION <description>]

[TOOLNAME <tool name>]

[<formal parameters>] // 参数说明

[<extended attribute list>]

END_APPLICATION

[<Workflow Application List>]

<generic tool id> : : = <identifier>

<tool name> : : = <string>

7.1.8 工作流相关数据

工作流相关数据定义了所有在工作流过程或者工作流模型当中用到的变量，因此这部分内容实际上也是一个变量表。

<Workflow Relevant Data List> : : =

DATA <data id>
[NAME <name>]
[DESCRIPTION <description>]
TYPE <complex data type>
[LENGTH <cardinal>]
[DEFAULT_VALUE <value>]
[<extended attribute list>]
END_DATA
[<Workflow Relevant Data List>]

<data id> : : = <identifier>

<value> : : = <initial> | <function access>

7.1.9 工作流参与者

工作流参与者是组织模型中的元素，他们或者执行工作流过程中的某一部分或者对其负责。工作流参与者的定义可能需要引入外部的组织模型，而且定义可以是多种不同的形式，我们称之为“类型”。具体的格式如下所示。

<Workflow Participant Specification> : : =
<Workflow Participant Declaration>
| <Workflow Participant Definition>

<Workflow Participant List> : : =
PARTICIPANT <participant id>
[NAME <name>]
[DESCRIPTION <description>]
<participant type description>
[<extended attribute list>]
END_PARTICIPANT
[<Workflow Participant List>]

<participant id> : : = <identifier>

<participant type description> : : = <declaration Ptype description>

| <definition Ptype description>

<Workflow Participant Declaration> : : = <Workflow Participant List>

<declaration Ptype description> : : = [TYPE <Ptype key>]

<Ptype key> : : = <ou key> | <hu key> | <ro key> | <system key>

<ou key> : : = ORGANISATIONAL_UNIT // 组织单元

<hu key> : : = HUMAN // 人

<ro key> : : = ROLE // 角色

<system key> : : = SYSTEM // 系统

7.2 一个 WPDL 的例子

在本节中，我们将用 WPDL 来定义一个实际的经营过程。在这个经营过程中，基本包含了上文所介绍的工作流模型的标准实体与属性。当把这些内容全部用 WPDL 进行描述时，所用的格式也将完全符合工作流管理联盟的标准。通过这个实际的例子，读者将对如何使用 WPDL 定义工作流有一个更深的了解。

7.2.1 过程描述

FBN 体育用品公司是卢森堡的一家专门制造与销售全套足球、篮球、网球以及田径运动装备器材的公司。他们的销售对象主要是批发商、零售商以及分销商。所有产品的销售都是通过订单来进行的。目前公司已由原来的小型企业逐步发展壮大，但同时也出现了一些问题，比较突出的一个就是对客户的响应时间变得越来越长。公司决定实施工作流管理系统，以便改进自己的关键业务流程。

在目前的环境下，所有的信件都是首先进入公司的信件室进行分检，然后再送至各个部门的，包括销售部、财务部、生产制造部门以及经理办公室，如图 7.2 所示。

由于公司 80%的订单都是以信件或传真的方式传递的，因此公司决定在信件室中使用扫描仪来增强这一关键部门的处理能力。在改进的新流程中，信件室的职员被授权允许审阅所有需要扫描的信件，并根据不同的类型传递到不同的部门：

如果是订单，则并行地发送到销售部、财务部及生产制造部门；

如果是货款或者货款的发票，则发送到财务部；

如果是标明“私人”或者“经理”的信件，则将直接被送到个人或者经理秘书手中。

进一步地，在每一个部门中都有相应的处理流程，在这里不再赘述，读者可以参考工作流管理联盟的相关文档，文档号为 WfMC TC-1016-X。

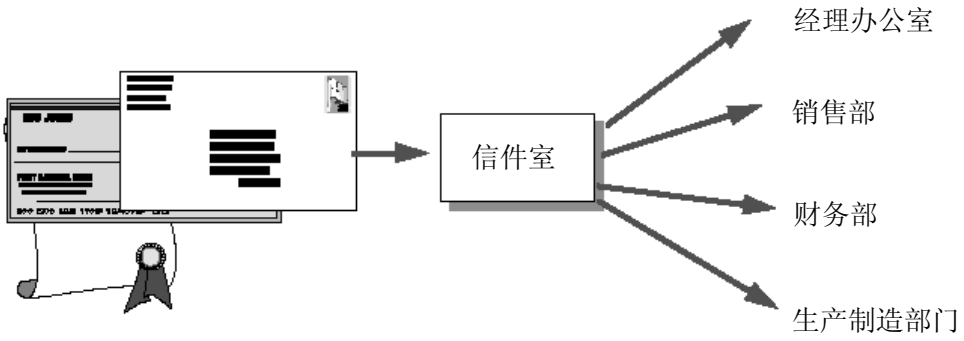


图 7.2 FBN 的信件分检过程

7.2.2 工作流模型的 WPDL 描述

```
MODEL                  'Representative_Business_Example'
    WPDL_VERSION       "7.0 Beta版"
    VENDOR             "Vendor A"
    CREATED            1998-07-15
    NAME               "一个经营过程的示例"
    DESCRIPTION        "通过本例来演示WPDL的基本用法"
    AUTHOR             "清华大学CIMS ERC"
    STATUS             UNDER_REVISION
    EXTENDED_ATTRIBUTE 'VENDOR_A_SPECIFIC' STRING
                       "这是 Vendor A 的一个扩展属性"

    // Workflow Participant List 工作流参与者列表
    PARTICIPANT        'France_Billy'
    NAME               "France Billy"
    DESCRIPTION        "经理秘书"
    TYPE               HUMAN
    END_PARTICIPANT
```

PARTICIPANT 'Tim_White'

NAME "Tim White"

DESCRIPTION "信件室职员"

TYPE HUMAN

END_PARTICIPANT

PARTICIPANT 'Presidents_Secretary'

NAME "经理秘书"

DESCRIPTION "处理经理的信件"

TYPE ROLE

END_PARTICIPANT

PARTICIPANT ' Mail_Room_Clerk '

NAME "信件室职员"

DESCRIPTION "处理所有的公司信函"

TYPE ROLE

END_PARTICIPANT

PARTICIPANT 'VP_Sales'

NAME "销售副经理"

DESCRIPTION "主管销售部门的副经理"

TYPE ROLE

END_PARTICIPANT

PARTICIPANT 'Sales_Department'

NAME "销售部"

DESCRIPTION "负责处理客户定单"

TYPE ORGANISATIONAL_UNIT

END_PARTICIPANT

PARTICIPANT 'Manufacturing_Department'

NAME "生产制造部门"

DESCRIPTION "生产制造相关产品"

TYPE ORGANISATIONAL_UNIT

END_PARTICIPANT

PARTICIPANT 'Finance_Department'

NAME "财务部"

DESCRIPTION "财务收支"

TYPE ORGANISATIONAL_UNIT

END_PARTICIPANT

PARTICIPANT 'Customer_Support'

NAME "客户支持部"

DESCRIPTION "处理所有客户咨询"

TYPE ORGANISATIONAL_UNIT

END_PARTICIPANT

// **Workflow Application List** 工作流应用列表

APPLICATION 'scan_document'

NAME "扫描文档"

TOOLNAME "winscan.exe"

OUT_PARAMETERS 'scanned_document'

END_APPLICATION

APPLICATION 'identify_document'

NAME "识别文档"

IN_PARAMETERS 'scanned_document'

OUT_PARAMETERS 'document_type'

END_APPLICATION

APPLICATION 'send_document'
NAME "发送文档"
TOOLNAME "abcmail.exe"
IN_PARAMETERS 'scanned_document'
'document_type'
END_APPLICATION

APPLICATION 'handle_document'
NAME "处理文档"
IN_PARAMETERS 'scanned_document'
'document_type'
END_APPLICATION

// **Workflow Process Relevant Data List** 工作流过程相关数据列表
DATA 'document_type'
TYPE STRING
NAME "文档类型"
DEFAULT_VALUE "客户定单"
END_DATA

DATA 'scanned_document'
TYPE REFERENCE
NAME "被扫描的文档"
END_DATA
// <插入“信件室”的工作流过程'In_the_Mail_Room'>
// <插入“财务部”的工作流过程'At_the_Finance_Department'> 略
// <插入“生产制造部门”的工作流过程'At_the_Manufacturing_Department'> 略
// <插入“销售部”的工作流过程'At_the_Sales_Department'> 略

END_MODEL

在以上的 WPDL 定义中，完成了对模型的头信息、工作流参与者、工作流应用以及工作流相关数据的定义。这些数据将在后面具体的工作流过程当中使用。

7.2.3 信件室的处理过程

上文中我们已经对信件室的处理过程进行了初步地描述，为了直观起见，我们用自己开发的工作流建模工具CIMFlow对这个过程进行建模，如图7.3所示。

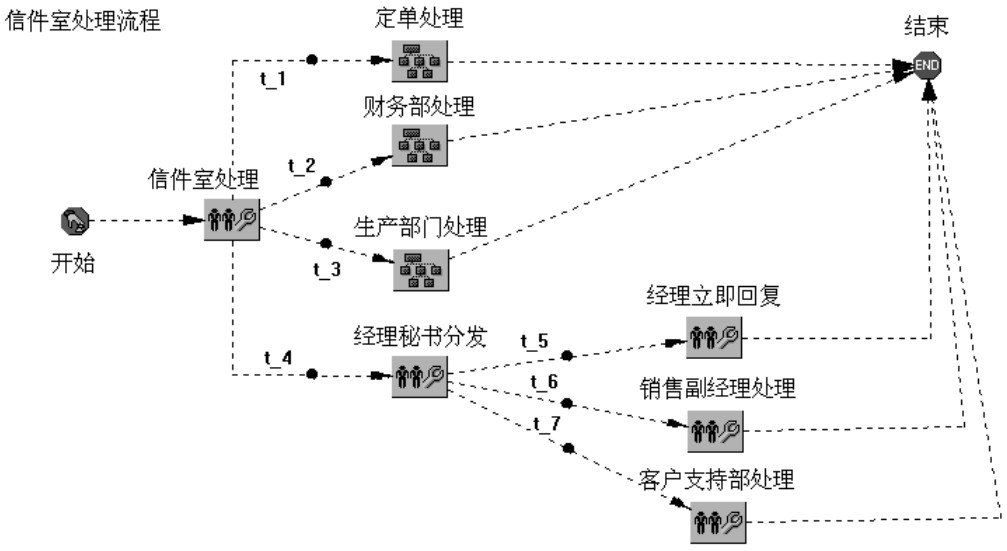


图7.3 信件室处理流程图示

这样的一个过程将作为整个工作流模型的一部分，被插入到前面已定义好的WPDL模型当中的<插入“信件室”的工作流过程'In_the_Mail_Room'>的注解之后，如下所示。

```
WORKFLOW                'In_the_Mail_Room'
    CREATED             1998-07-15
    NAME                "信件室处理流程"
    // Activity List  活动列表
    ACTIVITY            'MailRoom'
        NAME            "信件室处理"
        IMPLEMENTATION  APPLICATIONS
                        TOOL_LIST
                            'scan_document'
```

```
                                'identify_document'
                                'send_document'
                                DESCRIPTION "顺序执行以上应用"
                            END_TOOL_LIST
    PERFORMER       'Mail_Room_Clerk'
    SPLIT           AND
END_ACTIVITY

ACTIVITY            'Sales_Order_Handling'
    NAME            "定单处理"
    IMPLEMENTATION  WORKFLOW SYNCHR 'At_the_Sales_Department'
    PERFORMER       'SALES_DEPARTMENT'
END_ACTIVITY

ACTIVITY            'Finance_Processing'
    NAME            "财务部处理"
    IMPLEMENTATION WORKFLOW SYNCHR 'At_the_Finance_Department'
    PERFORMER       'Finance_Department'
END_ACTIVITY

ACTIVITY            'Manufacturing'
    NAME            "生产部门处理"
    IMPLEMENTATION
            WORKFLOW SYNCHR 'At_the_Manufacturing_Department'
    PERFORMER       'Manufacturing_Department'
END_ACTIVITY

ACTIVITY            'Presidents_Secretary_Dispatch'
    NAME            "经理秘书分发"
    IMPLEMENTATION  APPLICATIONS 'send_document'
```

```
    PERFORMER       'Presidents_Secretary'
    SPLIT           XOR 't_5' 't_6' 't_7'
END_ACTIVITY

ACTIVITY            'Immediate_Response_by_President'
    NAME            "经理立即回复"
    IMPLEMENTATION  APPLICATIONS 'handle_document'
    PERFORMER       'Presidents_Secretary'
END_ACTIVITY

ACTIVITY            'VP_Sales_handling'
    NAME            "销售副经理处理"
    IMPLEMENTATION  APPLICATIONS 'handle_document'
    PERFORMER       'VP_Sales'
END_ACTIVITY

ACTIVITY            'Customer_Support'
    NAME            "客户支持部处理"
    IMPLEMENTATION  APPLICATIONS 'handle_document'
    PERFORMER       'Customer_Support'
END_ACTIVITY
// Transition Information List 转移信息表
TRANSITION          't_1'
    FROM            'MailRoom'
    TO              'Sales_Order_Handling'
    CONDITION       'document_type'="客户定单"
END_TRANSITION

TRANSITION          't_2'
    FROM            'MailRoom'
```

```
    TO              'Finance_payment_processing'
    CONDITION       'document_type'="客户定单" OR
                    'document_type'="付款发票" OR
                    'document_type'="货款"
END_TRANSITION

TRANSITION          't_3'
    FROM            'MailRoom'
    TO              'Manufacturing'
    CONDITION       'document_type'="客户定单"
END_TRANSITION

TRANSITION          't_4'
    FROM            'MailRoom'
    TO              'Presidents_Secretary'
    CONDITION       'document_type'="经理信函"
END_TRANSITION

TRANSITION          't_5'
    FROM            'Presidents_Secretary_Dispatch'
    TO              'Immediate_Response_by_President'
    CONDITION       'document_type'="要立即回复的信函"
END_TRANSITION

TRANSITION          't_6'
    FROM            'Presidents_Secretary_Dispatch'
    TO              'VP_Sales_handling'
    CONDITION       'document_type'="销售报告"
END_TRANSITION
```

```
    TRANSITION          't_7'
        FROM            'Presidents_Secretary_Dispatch'
        TO              'Customer_Support'
        CONDITION       'document_type'="产品咨询"
    END_TRANSITION
END_WORKFLOW
```

WPDL 作为工作流管理联盟所推行的一种标准的工作流描述语言，已经在多家厂商的工作流系统中得以支持。WPDL 使得在不同的工作流系统之间交换模型定义成为可能。为了与国际标准结合，我们在所开发的工作流建模工具中也加入了“导入”与“输出”标准的 WPDL 格式文件的功能。“导入”是把标准的 WPDL 文件转换成我们定义的模型，并以图形化的方式显示出来，如上文中图 7.3 所示的模型；“输出”则是把用户通过 CIMFlow 的图形化界面而建立的工作流模型转换成为文本格式的 WPDL 文件。这样用 CIMFlow 建立的工作流模型就与任何支持 WPDL 的工作流建模工具兼容，也可以作为所有符合 WfMC 规范的工作流执行系统、BPR 工具的模型使用。

第 8 章 工作流的分布与实现技术

一旦工作流模型确定后，如何快速高效地实现工作流系统的执行就成为设计人员必须考虑的问题。描述企业业务过程的工作流模型除了对其进行分析外，还需要在实际应用中运转才能更好地发挥它的实际效益，否则可能总是停留在“纸上谈兵”阶段。在这里需要指出的是，工作流模型与工作流系统的实现方案并不构成一一对应的关系，也就是说，对应于同一个工作流模型，可能有多种不同的系统实现方案；同样，一种系统实现方案也可以执行多个工作流模型。工作流模型实现了系统对工作流过程的描述，而实现方案则决定了系统自身的运行机制，二者具有一定的关系，但不能相互确定。

确定一个工作流系统的实现方案主要需要考虑两个方面的内容：（1）首先是选择系统所采用的底层通讯基础结构，这一基础结构将关系到系统中的各个组成部分之间以什么方式来进行连接，这个基础结构是分布式应用赖以生存的基础。比较典型的基础结构有远程过程调用 RPC、面向对象的分布计算环境 CORBA、基于 TCP/IP 的 Web、各类消息传递系统以及代理系统等。（2）要确定系统各组成部分之间的协作过程。从模型的提交、运转到监控涉及多个软件模块间的协作，这些具有不同功能且相互独立的模块之间所进行的互操作过程实现了工作流系统的运转。这部分工作主要包括接口定义、数据维护方式、操作处理过程等的确定。

本章从系统实现的角度来阐述涉及工作流系统执行的相关技术，并给出几个典型的系统实现方案供参考。

8.1 分布式工作流

从工作流技术所要解决的问题来看，它必然要以分布的形式出现。因为无论是从企业的信息环境、组织环境，还是与外界的协作环境来看，都具有着明显的分布式特点：网络的延伸，系统的异构，人员的分散，供求关系的全球化等等。在这样的环境下要完成不同应用系统的集成、不同组织人员的协作并最终达到实现经营过程运作的自动化与高效率，所采用的工作流管理系统必然要具有分布式的特点。

“分布式工作流”的概念是相对于早期的集中式工作流管理系统而言的。在工作流技术发展早期的 20 世纪 80 年代初，工作流系统主要表现为应用于某些特定领域的、相对独

立的应用系统，比如图像、文档管理系统。它们以文件共享的方式实现任务间的协作。这种“集中式”的情况反映了当时工作流技术的状况。进入 20 世纪 90 年代，随着计算机与网络技术的迅速发展，特别是在 Internet 应用日益普及的情况下，现代企业信息系统的分布性、异构性和自治性的特征越来越显著，相应的企业信息资源也分布在异构的计算机环境中，信息源之间的连接表现出松散耦合的特点，这样的信息系统环境简称 HAD 环境（异构、自治、分布）。企业物理位置的分散性和决策制定过程的分散性特征日益明显，对日常业务活动详细信息的需求日益提高，Client/Server 体系结构和分布式处理技术（CORBA、WWW、OLE、JAVA）被广泛应用，以上这些情况都说明了这样一个事实——集中式信息处理的时代即将成为过去，取而代之的将是大规模的异构分布式信息处理与应用执行环境。在这种大规模的分布式环境下高效地运转相互关联的任务，并且对执行的任务进行密切监控已成为一种发展趋势。在这种客观技术背景下，工作流技术随之进入了一个新的发展阶段——分布式处理阶段。

从技术复杂性与实现的先后顺序上，工作流管理系统的分布可以分为以下三个层次：

1．工作流系统体系结构的分布

所谓工作流管理系统体系结构的分布是指从系统的层次上将工作流管理系统看成是一组相互协作的部分构成的。这些组成部分按照其完成的不同功能独立自成一体，不同的部分之间通过标准的互操作接口进行连接。工作流管理联盟（WfMC）在规范文档中提出了标准的工作流管理系统体系结构，这一体系结构充分体现了这种分布性，它将工作流管理系统划分成由过程定义工具、客户端应用、工作流管理工具、工作流机以及其他外部应用等几个部分组成，这些不同的组成部分之间通过 WfMC 定义的五类互操作接口进行连接。这些接口用来实现同一个工作流管理系统的不同组成部分之间以及不同工作流管理系统之间的互操作与集成。尽管不同系统的组成模块可以采用不同的组织方法、在不同的软硬件平台上实现，但这种整体结构上的分布性是相似的，这也正体现了工作流系统对客观环境的适应性。软件模块间的分布协作的本质是实现人的分布与协作。

2．工作流机的分布式执行

工作流机是工作流管理系统提供执行服务的核心模块，它的分布是在系统体系结构分布的基础上所实现的更高层次上的分布。由于工作流机直接负责过程实例的解释和执行，它的性能将直接影响到整个系统的运转效率。在集中式工作流机的情况下，是由一个工作流机来控制所有计算机上活动的执行，这种集中式的工作流机处理方式在系统的可靠性、

可扩展性、实用性以及吞吐量等方面都不能满足企业执行大规模复杂应用的需求。分布式工作流机是指采用一组分布在不同节点上的工作流机来共同协作完成对整个工作流实例的执行，每个工作流机完成其中一部分实例的执行，不同的工作流机之间通过可靠的通信机制实现协作。通过由分布在不同网络节点上的多个工作流机协作来运行工作流过程可以明显地改善集中式工作流机的性能瓶颈问题。

当然，工作流机的分布也带来了一些需要解决的新问题，比如工作流机的分布导致系统的复杂性增加、数据的一致性维护相对困难等等。目前虽然已经出现一些具有分布式工作流机的工作流管理系统（包括产品与原型系统），但在实际性能上还不完全令人满意，这方面的技术还有待于进一步的发展、成熟。

3. 工作流模型的分布式定义与柔性执行

工作流模型的分布是指在一个分布的环境下由参与人员协作完成工作流模型的定义。这完全是相对于集中式的工作流定义而言的。在集中式工作流定义方式中，工作流模型的建立一般采用由顶向下的递阶建模方法，工作流模型由工作流定义人员（通常是熟悉企业业务的流程设计师）采用一个工作流定义工具或者 BPR 工具，按照逐步分解、逐步细化的方法完成。完成这项定义工作的前提条件是这个（或者这些）人员熟悉全部的企业业务过程，并且这些业务过程对它们都是开放的（不存在保密问题）；在组织结构上，所有这些业务过程都在一个统一的组织结构管理下。这些前提条件对于一个企业而言是完全能够成立的。但是如果所建立的工作流模型描述的不是一个组织结构下的流程，比如描述一个虚拟企业结构下的涉及多个合作伙伴企业的产品设计制造过程时，上述前提条件就不一定成立。首先，要将这些伙伴企业的相关业务过程全面了解清楚是一件费时又费力的工作，而且一般说来，一个企业的业务流程（尤其是关健业务流程）在很大程度上是属于一个企业的商业秘密，它不会因为加入了一个战略的动态联盟而全部对外开放，另外由于不在同一个组织结构的管理下，各个伙伴单位处于一种协作的对等关系，所以建立一个全局的工作流模型也不十分的必要，因为没有任何一个组织能够对整个流程进行管理。在这种情况下，比较可能的实现方法就是由不同的伙伴企业各自完成其内部流程模型的建立，这些内部模型对外公开其交互的接口，以便实现不同企业工作流模型之间的连接。这样所建立的工作流模型就不完全是一个自顶向下的分解过程，而是由底向上综合和自顶向下分解二种方法的组合。自顶向下的分解过程完成整个跨企业流程的分解，在各个伙伴企业完成了其内部流程建模后，采用由底向上的综合方法进行模型的连接和匹配，从而形成整个的工作流模型。

这样形成的工作流模型就是一个典型的分布式模型，因为大家共同了解的，或者共同承诺的是模型之间的交互接口，不同部分的模型细节属于企业的商业秘密由各个企业分别进行维护。

分布式建模方法的难点问题是模型的集成问题，即如何由许多部分模型形成一致的全局模型。为了实现这些模型的集成就需要规定清楚模型之间的接口都需要提供什么信息，这些信息之间交互采用什么样的协议或者协调方法来完成。

工作流模型的柔性执行是与工作流模型分布密切相关的另外一个问题。在目前的系统中，工作流建模工具（过程定义工具）是用户建立工作流模型的唯一工具，工作流要想运行，必须先由建模人员经过专用的建模工具来定义工作流模型，即由用户完成集中式的工作流模型定义。这种定义→提交→运行的方式大大限制了工作流系统在企业中的实际应用，因为企业中存在着大量临时决定的、非结构化的经营过程，它们往往需要尽可能迅速地付诸实施，这样执行前的集中式定义就成为一种低效的过程。另外，在集中式的模型定义方式下，如果用户想对某一已经提交了的工作流实例中的活动做出及时的修改，也是一件很困难的事情。这导致工作流系统运行时的柔性大大降低，其根源就在于模型定义的集中化、模型定义与实例运行这两个阶段的严格分离。工作流定义的分布就是把一部分（或全部）的工作流定义功能加入到运行时的客户端应用中，以使处于执行阶段的用户可以自己定义后续的活动，实现“边定义一边执行”的工作流运行方式。这样，工作流执行服务就可以被每一个有定义权限的用户随时使用，最大限度地提高了系统的柔性与效率。目前，实现这一层次的分布正是许多厂商与研究机构所努力的方向。

从系统实现的角度来看，分布式工作流技术主要涉及以下几个方面：

1） **消息传递（底层通讯技术）**

消息传递可用于在工作流机之间、工作流机与用户之间进行信息的交互，是分布式的应用组件实现互操作的一种方式。通常来讲，消息传递是工作流管理系统底层通讯基础的一种形式，它可以是特定格式的消息，也可以是符合标准的电子邮件。在现有的工作流产品中有一大类就是基于消息的系统，这种类型的产品都实现了与一种或多种消息传递系统（电子邮件系统）的集成。

2） **CORBA**

CORBA 是对象管理组织（OMG）提出的一种分布式对象计算的标准规范，用于实现异构平台上的分布式应用开发以及不同应用系统间的集成。基于这一规范开发的应用程序无需考虑底层的网络协议和数据传输，而是依靠对象请求代理（ORB）机制来实现对象间

的通信与激活。处于分布状态的各个子系统被封装于不同的 CORBA 对象之中，通过与实现语言无关的接口定义语言（IDL）来定义与外界进行交互的接口，再经由编译器映射到不同的实现语言上。对象请求代理（ORB）则负责将客户方提交的请求发送给服务方，并将服务方返回的结果传给客户方。

CORBA 作为一种抽象的规范定义并不限制具体的实现方案，这一点对于软件供应商而言最具吸引力，目前市场上已经有多种 CORBA 的实现产品，这为用户的选择提供了方便。由于 CORBA 是面向对象的，这就意味着面向对象的种种方便与强大功能将在 CORBA 的使用中得以体现，如系统的开放性、可重用性以及与原有系统的无缝集成和新功能的快速开发等等，这将有助于简化分布式应用的集成、开发过程。另外，随着 CORBA 的不断完善，OMG 还将在 CORBA 服务中提供有关工作流方面的支持。这些优点都将使 CORBA 成为用户实现企业级工作流解决方案的一种可能的选择，许多研究机构已经或者正在开发基于 CORBA 的工作流管理系统。

与 CORBA 共存的另外一种对象模型体系就是微软的分布式组件对象模型（DCOM），DCOM 的前身是微软的 OLE，它也是一种以面向对象的方法解决分布、异构信息源集成问题的方案，但由于它并不是供应商独立的（即用户需要完全依赖于微软），而且在非 Windows 平台上具有很大局限性，所以在企业级的应用方面明显不如 CORBA。

3） **WWW**

WWW 是 Internet 最普遍的一个应用，通过 Web 浏览的方式可以使人们能访问到世界各地的信息，而且使用十分方便，只需要安装前端的一个浏览器即可。随着 Web 技术的不断发展，特别是动态网页技术的成熟，Web 不仅能够提供静态信息，而且能够实现与后台数据库的集成，这使得 Web 成为一种很有力的分布式交互方式。在企业的工作流系统中，Web 可以用来代替传统的面向执行者的客户端应用，使用户无需安装特定的软件，通过浏览器就可以获得自己的任务表，并能够进行任务的提交等功能。目前基于 Web 的工作流系统已成为一种流行趋势，许多供应商纷纷开发新产品或者在原有产品的基础上增加对 Web 的支持。

4） **Java**

Java 是 Sun 公司于九十年代推出的新型面向对象编程语言，在很短的时间内，它便成为 Internet 应用开发的流行语言。它将面向对象、平台无关性、鲁棒性、安全性、多线程等诸多特性集于一身，同时代表了一种开放、自由、创造的新的应用设计思想，为用户提供了一个良好的编程环境。尤其是在网络应用方面，它能够实现与 Web 的紧密集成，使网

页更具丰富的交互能力；而且通过字节码运行在 Java 虚拟机上，很容易地实现了“一次编码，到处运行”的跨平台特性。这便又为异构平台上的分布式应用开发提供了一种可以选择的方案。目前已经有工作流产品和原型系统采用 Java 做为开发语言。

5) **分布事务处理技术**

事务的概念来自于数据库研究领域，主要是为了实现数据在操作中的一致性。与此相类似，工作流也需要引入相应的事务概念，来保证工作流数据在工作流分布执行过程中的正确性。从实际应用的角度来看，具有分布事务处理功能对于工作流系统而言是必不可少的。人们希望通过研究工作流的事务特性，将高级事务模型（ATM）与工作流管理技术相结合，用良好定义的模型语义与恢复机制来更好地支持企业的经营过程，从而提高工作流系统的可靠性与实用性。从目前的工作流产品与研究成果来看，在这方面尚有待于进一步研究。

8.2 工作流系统的底层基础结构

不同的硬件平台、操作系统、数据库、网络协议等并存于同一个系统中，这就是分布计算中常说的异构性问题。比如，一个典型的企业内部网络包括有大型主机、UNIX 工作站和 PC 机，各种机器所采用的操作系统和网络通信协议也是千差万别的。在理想的情况下，人们希望能够使用最好的硬件和软件组合来支持企业或企业部门有效地运作，但异构性的存在阻碍了应用系统之间实现良好的互操作，使得这种愿望难以实现。因此，很多企业都不得不把很大的精力和财力放在处理底层系统的异构性方面，而分散了对企业经营过程直接相关的应用系统的注意力，也限制了企业选择应用系统的范围。同时，应用系统的可移植性、可重用性和互操作性等也受到了大大的影响，这一方面增加了费用、延长了实施周期；另一方面也阻碍了企业实现过程集成。

在异构环境下实现信息和软件资源的共享是一项极大的挑战，而一个开放的标准是解决此类问题的关键。同时，对应用软件的可扩展性和可重用性的要求使得面向对象的分析、设计和编程技术得到了广泛的运用。在这两个因素的驱使下，对分布对象计算（Distributed Object Computing，简称为 DOC）技术和标准的研究制订成为八十年代末、九十年代初计算机技术发展的一个热点。

国际性行业协会组织“对象管理集团（OMG）”是一个非盈利性组织，成立于 1989 年，至今已拥有超过 700 名成员。OMG 组织的中心任务就是基于实用的对象技术，建立一个体

系结构和一组规范，在分布式环境下实现应用的集成，使得基于对象的软件成员在分布异构环境中可重用、可移值和可互操作。公共对象请求代理体系结构 CORBA（Common Object Request Broker Architecture）就是由 OMG 组织制定的这样一个工业规范。

8.2.1 对象管理参考模型与 CORBA 体系结构

OMG组织制定了分布计算的参考模型，称为对象管理参考模型（OMA：Object Management Architecture），其体系结构如图8.1所示：

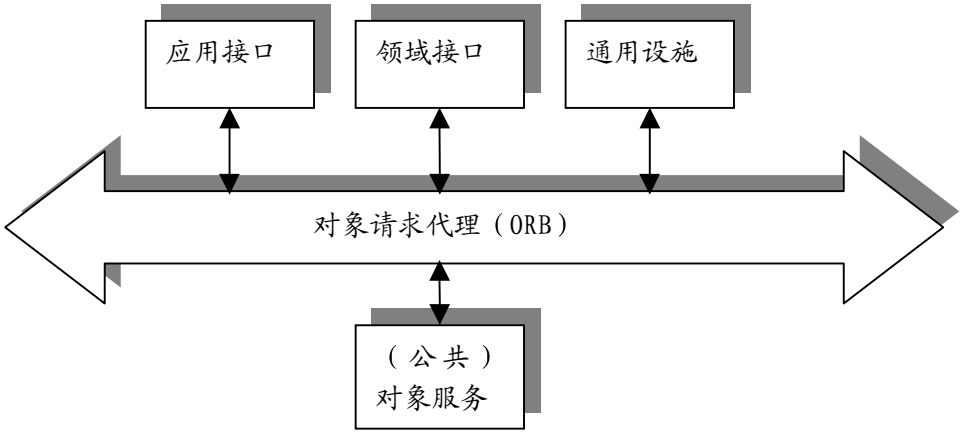


图 8.1 对象管理参考模型 OMA 体系结构

OMA 体系结构的核心是对象请求代理（ORB：Object Request Broker），CORBA 规范对 ORB 的组成和功能进行了定义，它支持对象服务、通用设施、领域接口和应用接口之间的交互和通信。

在 OMA 参考模型的接口层示意图中，对象服务（Object Services）是独立于应用领域、为各种分布式对象软件提供的一组基本服务的接口，如命名服务，事件服务等；通用设施（Common Facilities）是向终端用户应用提供的一组服务接口，如组合文档等；领域接口（Domain Interfaces）是针对某个应用领域（如产品数据管理 PDM）而提供的服务接口；应用接口（Application Interfaces）是特定的高层应用的对外接口。

对象请求代理 ORB(Object Request Broker)是 CORBA 的对象互操作中介，它作为应用对象间服务请求 / 响应的中间代理，接受对象请求并把请求转给相应的对象，服务完成后又把执行结果或异常返回给请求者。ORB 可以使对象以语言、位置和平台独立的方式发出请求和提供服务，相互协同工作，从而建立真正的分布处理，是实现分布对象互操作的核心。

CORBA ORB 的组成结构如图 8.2 所示：

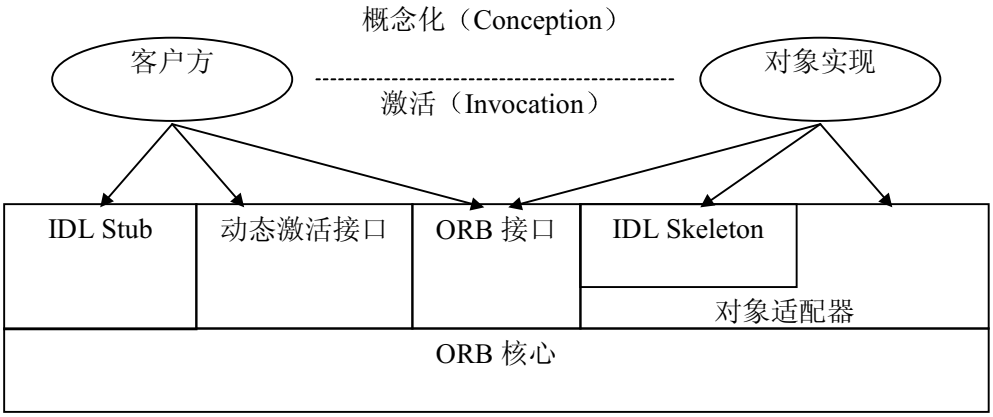


图 8.2 CORBA ORB 结构

1） 客户方（Client）通过 ORB 向对象实现发出服务请求；

2） 对象实现（Object Implementation）是提供服务的对象（服务方），含有实现服务的对象的数据和代码；

3） 对象请求代理 ORB 为客户方提出的请求进行对象实现定位，并通过对象适配器激活对象实现；

4） 动态激活接口 DII（Dynamic Invocation Interface）为客户方提供动态构造服务请求的方法，适用于访问未知服务的情况；

5） 对象适配器（Object Adapter）接到 ORB 的消息后激活相应对象实现，把请求传递给它，并向对象实现提供 ORB 服务；

6） IDL Stub：提供客户方访问对象服务的静态接口。它是由 IDL 编译器产生的。

7） IDL Skeleton：提供对象实现服务的静态接口，它是由 IDL 编译器产生的。

在 CORBA 体系结构中，ORB 在应用对象间建立联系，实现应用对象间的服务请求、结果发送等功能。应用对象之间是动态的客户 / 服务器（Client/Server）关系，一个应用对象既可以是服务方，也可以是客户方，甚至可以同时是服务方和客户方。在每一次互操作过程中，客户方和服务方相对独立，它们只和 ORB 进行交互，发出请求或获得服务结果。

ORB 提供给用户一种全局透明的访问能力，它隐藏了如下细节：

1） 对象的位置：即客户方无需了解自己的服务方具体位于何处，它可能就在本地机器上，也可能是在相距甚远但是有网络联结的异地端；

2） 对象的实现：即客户方无需知道为自己提供服务的服务对象具体是如何实现的，包括用何种语言编写，运行的操作系统是什么，是基于何种硬件平台等等；

3） 对象的状态：即客户方在发送请求时，不需要知道服务对象是否正处于激活并准备就绪状态，ORB 会在必要时提前使服务对象做好准备。

正由于 ORB 所具有的以上特征，使用户原本难以跨越的异构平台变得十分透明、顺畅，他们可以集中精力只关心上层的应用领域内的问题，而无需理会系统的异构、分布等底层细节了。因此，ORB 可以说是 CORBA 体系结构的基础，它使得异构系统之间的互操作性成为了可能。至于如何操作，则要由用户通过 CORBA IDL（接口定义语言）来实现。

CORBA IDL 是一种与编程语言无关的接口定义语言，用来定义对象的请求／服务接口，描述应用对象所封装的内容及界限。它类似于 C++中类的描述，也包括属性和操作两部分，并且也支持接口之间的继承，以实现对象的可重用性。IDL 定义经过编译后成为可为开发人员直接使用的头文件和 stub 程序。由 OMG IDL 到任何编程语言的映射理论上都是可以得到支持的。就目前而言，供应商们已实现了诸如对 C、C++、Smalltalk 等常用编程语言的映射，对 Java 的支持也日益成熟。

CORBA 为应用系统集成提供了良好的解决办法。通过引入 ORB，CORBA 解决了异构环境下的应用间互操作问题，应用系统可以实现位置和底层平台透明的对象访问。同时，CORBA 提供的 IDL 接口定义标准又能解决应用软件接口不一致的问题，使得我们能够通过在应用开发中采用该接口定义标准或为原有应用扩充标准接口的方式实现各种应用的集成。

CORBA 作为系统集成的一种工业标准体系结构，其主要优点在于：

1） CORBA 简化了分布式应用的集成，对于最终用户而言，它更易使用，因而在时间和成本方面都有所节约。

2） CORBA 作为一种抽象的规范定义并不限制具体的实现方案，这一点对于软件供应商而言最具吸引力。因为这种灵活性使供应商可以充分利用现有的网络设施，比如有些供应商（如 IONA）是以基于 ONC（Open Network Computing）兼容的 RPC 方式实现 CORBA，而有些供应商（如 HP）是利用 OSF DCE 来实现 CORBA，还有些供应商（如 SunSoft）则跳过 RPC 层直接在底层直接实施 CORBA，大多数供应商都提供使用 OMG IDL 接口实现连接库代码的支持。这种软件结构与实现手段相分离的特点，使得供应商们可以先利用 IDL 完成软件的结构设计，然后再选择合适的通信机制，以使系统具有最大限度的可用性。

3） 与原有的基于 RPC 机制的单纯的 C/S 结构相比，CORBA 结构更有利于资源的灵活、合理利用。因为 CORBA 是对等式的分布计算环境，所有应用对象之间的地位是平等

的，其担任的角色也是可以转换的：当某一对象产生服务请求时就被称为客户方，而当它接受服务请求时就被称为服务方。绝大多数 CORBA 对象都可以担任客户方和服务方两种角色。

4） CORBA 是面向对象的，这意味着面向对象的种种方便与强大功能将在 CORBA 的使用中得以体现，如系统的开放性、可重用性以及与原有系统的无缝集成和新功能的快速开发等等。

5） CORBA 作为一种标准，其核心元素的稳定性是有保证的。CORBA 产生于拥有 700 多成员的 OMG 组织，该组织包括了多家主要的计算机软硬件厂商及大的科研院所，并得到 X/Open，OSF，COSE，CI Labs，X/Consortium 等的支持，权威性是毋庸置疑的。自 1991 年 CORBA 1.1 版本的问世以来，CORBAR 的功能不断扩展，对异构平台的兼容性不断增强，并且对诸如 Java 等新技术也提供支持。

8.2.2 CORBA 的应用状况

CORBA规范自诞生以来，由于其目的正是制定标准以实现在不同的网络和计算机环境下独立开发的应用软件的互用性，保证组件/部件的重用性、可移植性和互操作性，基于已商业化了的对象技术，使分散的应用软件的集成成为可能，因此，得到了众多厂商和机构的支持，包括：IBM、HP、SUN、DEC、APPLE、Oracle、Sybase和麻省理工学院、华盛顿大学等大公司、大学和科研部门。

CORBA 为解决企业集成提供了规范。它提供如下支持：从通用的台式应用系统中存取分布信息和资源；使已有的企业数据和系统作为网络资源来利用；通过集成企业的特殊的应用和功能，增强通用的台式应用系统和工具；支持软件可重用，集成新的技术和新的资源。

从 90 年代初期至今，出现了许多成功的 CORBA 实现系统，如 IONA 公司的 Orbix 系统、Visigenic 公司的 VisiBroker 系统、IBM 的 SOM System Object Model 及 DSOM（Distributed SOM）、SunSoft 的 DOE（Distributed Object Environment）、HP 的 DOMF（Distributed Object Management Facility）和 DEC 的 ObjectBroker 等。其中最著名的是 IONA 公司的 Orbix 系统和 Visigenic 公司的 VisiBroker 系统，它们在很多分布系统的开发和支持中（例如制造业、电信领域、金融领域、交通运输和计算机领域等领域）都得到了应用。例如：波音公司采用 Orbix 进行企业过程重组，VisiBroker 直接被 Netscape 的 Navigator 浏览器（4.0 及其以

上版本）所集成。

近几年来，随着应用需求的进一步发展，CORBA 技术的研究和发展进入了新的领域。

首先，随着 Web 应用的飞速扩展和 CORBA 技术的日益成熟，CORBA 的 Web 使能技术已成为近年来 CORBA 研究的热点之一。CORBA 与 Web 相结合的‘粘合剂’就是 Java 语言。

目前，国际上主流的 CORBA 产品均已实现了 Java 语言版本的 CORBA 系统，如 IONA 公司的 OrbixWeb，VisiGenic 公司的 VisiBroker 等。这些产品均采用了 CORBA 与 Web 相结合的解决方案，即 Web 浏览器通过下载 Java Applet 形式的 CORBA 客户方程序来访问 CORBA 应用服务。这已成为实现 CORBA 的 Web 使能的主要方式。由于 CORBA 与 Java 的结合日渐紧密并趋于融合，OMG 已在最新的 CORBA2.2 规范中正式提出 IDL—Java 映射；Sun 公司也在推出的 JDK1.2 中提供一个免费的 CORBA—Java 开发工具——Java IDL。

其次，在企业环境中，应用软件往往是基于不同的编程语言编制实现的，而且也往往分布于不同的操作系统上运行，不同的企业应用软件的功能和信息的集成是目前许多软件厂商和研究机构的重点和热点。CORBA实现了服务对象的位置透明性、实现透明性、执行状态透明性和通信机制透明性，只要按照OMG IDL接口描述语言对服务对象所提供的服务进行描述，客户系统和服务对象之间就可以透明地交互运行。CORBA的此种特性又被称为“软件总线”，应用软件或构件能够在“软件总线”之上实现“即插即用”（Plug & Play）。但是，仅依靠ORB还不能够行之有效地减轻（分布式）应用软件的开发、配置和管理的代价。例如，不同的软件系统之间在交互时通常需要对安全管理进行协商，有些软件系统可能还需要“事务性”的保证。这些功能ORB并没有提供，因此开发者及用户就必须以专有的方式去解决。

OMG组织提出的OMA体系结构在ORB之上定义了对象服务、通用设施和领域接口，正是希望通过将（分布式）应用软件的开发、配置和管理中的基本服务和功能规格化，从而实现这些服务和功能的可重用性，以降低开发、管理的开销。在对象服务、通用设施和领域接口的支持下，ORB的功能将更加完善和全面。

针对对象服务，OMG组织制订了“CORBAservices: Common Object Services Specification”规范，该规范简称为COSS。COSS规范由一组接口（Interface）和服务行为描述构成，其接口一般使用OMG IDL语言描述。目前的COSS规范主要包括以下的部分：名录服务（Naming Service）、事件服务（Event Service）、持久对象服务（Persistent Object Service）、生命周期服务（Life Cycle Service）、并行控制服务（Concurrency Control Servic

e）、外部化服务（Externalization Service）、关系服务（Relationship Service）、事务服务（Transaction Service）、查询服务（Query Service）、许可证服务（Licensing Service）、特征服务（Property Service）、时间服务（Time Service）、交易服务（Trading Service）和安全服务（Security Service）。由此可见，COSS规范几乎包含了分布系统和面向对象系统的各个方面；它的每个组成部分也都是非常复杂和全面的。

随着ORB的实现技术日益成熟，目前国内外对CORBA技术的研究也已经或正在逐步转移到公共对象服务的领域。一些知名的国际软件生产厂商，如IBM，Inprise，IONA，BEA和Siemens等，都对某些关键的基本服务和功能（如名录服务、交易服务、事件服务、事务服务和安全服务等）进行了较为长期的研究和开发，而且已经提供了一些商业产品并在许多应用领域得到了比较广泛的运用。其中比较著名的有：IONA公司的OrbixName（名录服务）、OrbixTrader（交易服务）、OrbixSecurity（安全服务）、OrbixOTS、OrbixOTM（事务服务）和OrbixTalk（事件服务）；Inprise公司的Visibroker名录服务和Visibroker事件服务等。

8.2.3 CORBA 与 DCOM 的比较

与 CORBA 共存的另外一种较为重要的分布式对象模型体系就是微软的分布式组件对象模型 DCOM（Distrubuted Component Object Model），DCOM 的前身是微软的对象链接与嵌入（OLE），它也是一种以面向对象的方法解决异构信息源集成问题的方案。在这里我们通过两者的对比，主要是想说明 CORBA 虽然具有如此多的优点，但也存在着一些不足之处，如果这些不足之处对于最终用户来讲恰恰是无法忍受的，那么 DCOM 也许正是合适的解决方案。

1） 对于跨平台特性而言，CORBA 具有着先天的优势。对于任何流行的操作系统，用户均可获得相应的 CORBA 产品支持，甚至于可以有多种不同厂商的产品可供选择。比如前面我们提到的 DEC 公司的 ObjectBroker，几乎涵盖所有的系统平台，包括 Mac OS、AIX、MVS、OS/2 Warp、 HP-UX、Windows 等。而 DCOM 最初是基于 Windows 平台（95 与 NT）的，在向其他平台的移植方面，仍有许多工作要做，但也只是一个时间问题。

2） CORBA 是建立于一系列的文档规范之上，是语言独立的，这保证了实现的灵活性，但同时也带来了很棘手的问题——不同厂商的 CORBA 产品之间很难进行互操作。这

一问题看上去很有趣：为了解决异构问题，结果却带来了新的异构需要解决。对于最终用户而言，虽然有很大的范围来选择不同厂商的 CORBA 产品，但是一旦选定，就有可能被迫只局限于某一家的解决方案，而很难再问津其他厂商了。直到 1994 年，OMG 发布了 IIOP（Internet Interoperable ORB Protocol）规范后，才使得这一状况有所改观，但问题并未完全解决：首先是 IIOP 在效率和性能上表现欠佳，另外就是各厂商执行 IIOP 规范的实际情况不得而知。所以 CORBA 产品本身的互操作性问题仍亟待解决。而与之相反，DCOM 是基于源代码的一种规范，是语言相关的，实施的核心源代码是兼容的，这就保证了 DCOM 产品之间具有良好的互操作性。

作为当前分布式对象模型中的两大主流，以 OMG 组织的 CORBA 和 Microsoft 公司的 COM（DCOM）分布式对象模型开发的分布式应用正在迅速增长。然而，正如当初计算机网络的发展一样，随着各种产品的发展和不断普及，产品之间的互操作需求也会不断增长，产品之间无法互通的矛盾甚至会制约产品和技术的进一步发展。对于 CORBA 和 COM（DCOM）来说，目前还无法评判孰优孰劣，也无法判断谁会在将来的竞争中取胜，最大的可能是两者在长时期内共同发展，这样使用不同模型开发的应用的集成就需要 COM(DCOM)/CORBA 的互操作；另一方面，两者各有其优势和特点，大多数异构环境将同时采用 CORBA 和 COM（DCOM）的混合环境，这种混合环境下应用的开发也会使用 COM/CORBA 互操作。因此，不论是当前还是将来，COM/CORBA 互操作都具有重要的意义。

众所周知，Microsoft 公司的 OLE（COM 和 DCOM 的前身）拥有众多的用户，因此 OMG 组织在颁布了 CORBA 标准 1.0 版本之后即投入 COM/CORBA 互操作研究，并于 1996 年 1 月 31 日由 DEC、HP、IONA、SunSoft 等公司联合提交了规范草稿（RFP）（COM/CORBA Interworking Part A），其内容稍做修改后并入 CORBA 规范 2.0 版本于 1996 年 7 月发布成为标准。这样做一方面用于指导 COM(DCOM)和 CORBA 的互操作的实现，使得各生产商可以按照该规范为用户提供转换机制相同的 COM(DCOM)/CORBA 互操作产品，另一方面也表明了 COM(DCOM)/CORBA 互操作的可行性，从而可以吸引更多的用户使用 CORBA，包括众多的 COM(DCOM)用户。

在制定互操作标准的同时，相关的互操作产品也相继出现，这些产品都是作为 CORBA 产品的一个组成部分提供给用户的，主要有 IONA 的 Orbix ActiveX，Sun 的 NEO，HP 的 ORBplus。

Microsoft 的 COM（OLE）只能在单机上运行，因此在 DCOM（分布式 COM，又称为

Networked OLE）出现以后 COM（DCOM）才成为完整意义上的分布式对象模型。DCOM 的出现无疑为 COM/CORBA 的互操作又增添了新的需求。目前 OMG 组织已经发布了由 BEA Systems、DEC、IBM、HP、IONA 和 NOVELL 等公司联合制定的 DCOM/CORBA 的互操作规范的征求意见稿及其修订稿（COM/CORBA Interworking Part B）。目前致力于实现 DCOM/CORBA 互操作的公司及其产品有：HP 公司的 ORB Plus、IONA 公司的 OrbixCOMet、Expersoft 公司的 CORBAplus ActiveX Bridge 和 DEC 的 ObjectBroker Desktop Connection (DTC)。与此同时，为了适应开放性、可扩展性和应用广泛性的需求，各个 CAD／CAM／PDM／MRPII 生产厂商软件产品基于 CORBA 技术和 Internet 技术的重构工作也正在如火如荼地进行。

8.2.4 消息传递系统、代理系统及 Web

除了 CORBA 以外，消息传递系统、代理系统及 Web 也可以做为实现工作流管理的底层基础结构，只不过它们并不象 CORBA 那样具有非常规范的标准，在具体的应用设计与开发的过程中，不同的系统可能会具有不同的特点。

消息传递是比较传统的一种分布式应用互操作方式，通过在不同的应用组件之间发送和接收消息来实现不同应用之间的相互响应。消息传递系统的代表性产品有 DEC 公司的 MessageQ、Novell 公司的 Tuxedo/Q、IBM 公司的 MQ Series 等，这些系统为上层的应用隐藏了复杂的通讯实现代码，并且屏蔽了操作系统、网络协议的异构性，它们一般通过 API 函数来为应用提供各种消息服务，因此很适合于连接分布式应用。

代理系统是 20 世纪 90 年代开始得到重视的一个研究方向，代理系统已经在许多不同的领域进行了应用，同样，它也被用来实现工作流管理系统。代理系统在对底层基本通信系统进行封装的同时，还可以对规划层等高层次的应用进行封装，它可以从较高层次上对用户的操作加以反映，并对用户屏蔽具体的实现方法，代理之间的协作完成过程的执行推进。代理一般还具有一定的智能化操作能力，比如自适应、自学习等功能，这些都有利于提高工作流系统的柔性与实用性。

Web 作为 Internet 的一个主要应用，在实现工作流系统方面也具有相当的优势。首先在价格方面，企业更容易拥有 Web 服务器或者与 Web 相连接，这是构建企业信息环境必不可少的一个基本要求；在用户的使用方面，具有瘦客户端特点的浏览器为用户提供了一个通用、友好且风格一致的界面，因此浏览器可以替代工作流系统中的客户端应用，这是

基于 Web 的工作流管理系统的一个极具吸引力的特点；在跨平台方面，Web 也可以比较容易地布置在多个计算平台上，因为底层的网络协议都是基于 TCP/IP 的。与其他类型的工作流管理系统相比，具有 Web 特征的工作流系统起步是比较晚的，它是以 Internet 的广为普及为前提的，但是这一类型系统的发展却十分迅速，目前已经成为一种比较流行的趋势，许多工作流厂商纷纷开发新的产品或者在原有产品的基础上增加对 Web 的支持，比如 Action 技术公司在原有的 ActionWorkflow 基础上推出的 ActionWorks Metro，还有 Ultimus 公司的 Ultimus 等。

8.3 几个典型的工作流系统实现方案

在本节中我们将介绍几种较为典型的工作流系统实现方案，它们分别出自于不同的研究项目，在一定程度上代表了几个不同的研究发展方向。虽然每个实现方案都已经具有其对应的工作流模型，但正如在 8.1 节中所述的那样，实现方案与工作流模型之间并非是紧密耦合的，设计人员可以参考这些实现方案，把它们应用于自己的工作流模型的运转当中。从应用的角度来讲，工作流系统的实现方案是具有普遍意义的。基于 CORBA 与 Web 的工作流系统实现方案，读者可参考 4.3 节中介绍的美国 Georgia 大学 METEOR 项目中的 ORBWork 与 WebWork，此处不再细说。

8.3.1 Exotica/FMQM

由于工作流本身所具有的分布性特征，要求相应的工作流管理系统能够实现对工作流模型的分布式执行功能。通常的基于客户/服务器方式设计的系统，往往是把有关过程执行的信息（工作流定义、工作流实例等）存放在网络上的某一个节点上，以便于监控、统计与同步。IBM Almaden 研究中心所进行的研究项目 Exotica 在工作流分布执行方面则提出了一种能够完全分布的执行模型。它通过持久消息（Persistent Messages）的方式来保存工作流相关执行信息，使得每一个执行节点都独立运行，工作流过程的执行不以某一个节点为中心，实现了完全分布。这种执行方式大大地提高了系统的可靠性、可扩展性以及柔性。

（1）持久消息

Exotica 的这种设计方案是建立在底层的消息传递系统之上的，类似的产品有 DEC 公司的 MessageQ、Novell 公司的 Tuxedo/Q、IBM 公司的 MQSeries。这些消息系统为上层的应用隐藏了复杂的通讯实现代码并且屏蔽了操作系统、网络协议的异构性，它们通过 API

函数为应用提供各项消息服务。这些产品的特点很适合于用来连接分布式应用，实现分布环境下的工作流管理功能。

在消息系统所构成的信息基础结构之上，分布的执行节点（即运行不同应用的网络节点）间是通过发送与响应消息来相互联系的。持久消息是指被保存在可恢复消息队列中的消息，它们一旦被放入消息队列，并且在相应的消息处理事务开始后，这条消息将一直被保存，直到处理完毕后，才从消息队列中取走。每一个执行节点都有一个（或多个）具有可恢复能力的消息队列，通过 PUT 与 GET 这样的函数调用来插入或取出一条消息。GET 调用可以取走队列中的任何一条消息，而不是仅能够取到位于队列首位的消息。另外，对于 PUT 与 GET 这样的操作，还要求它们具有事务特性，以保证当某些例外情况发生时，消息不会丢失。

（2）过程定义与节点绑定

在工作流建模阶段，工作流过程的定义是在安装了建模工具的节点上完成的。当一个过程定义完毕以后，这一定义将被整理成为对应于不同执行节点的过程信息。由于每一个执行节点只能够完成过程中的一个或几个活动，那么根据节点的能力信息，完整的过程定义就被分解成为多个独立的部分，这些独立的部分将被传送并保存在不同的执行节点上。

节点绑定的过程是通过运行在每一个执行节点上的节点管理器（Node Manager）来完成的。节点管理器接受来自于建模工具的已被整理过的过程信息，并在本地生成过程表。过程表所保存的内容包括过程标识符、活动标识符、输入与输出的控制连接弧表、开始与结束条件、输入与输出的数据连接弧等。

当节点管理器在本地生成过程表后，它就启动一个线程用以管理该过程表所对应的过程实例，这一线程被称为“过程线程”。过程线程的第一个任务就是产生消息队列以接收由其他节点发出的消息，然后过程线程不断检查队列中的消息并决定活动实例是否能够执行。如果某一活动实例可以被执行，那么过程线程就生成一个新的“活动线程”来管理这一活动实例的执行，活动线程将把活动实例的相关信息保存在一个实例表中。实例表的内容包括过程标识符、实例标识符、输入/输出数据、活动状态等。

由一个工作流定义所生成的所有实例在一个执行节点上都对应着同一个持久消息队列，一个执行节点能够同时维护多个消息队列以响应不同类型的工作流实例。

图 8.3 是一个示例，首先在定义节点上建立过程 A 的模型，然后过程 A 的模型被分解，并发送到三个执行节点 1、2、3，最后由三个执行节点 1、2、3 完成过程 A 的执行，各个节点的结构均表示在图 8.3 中。

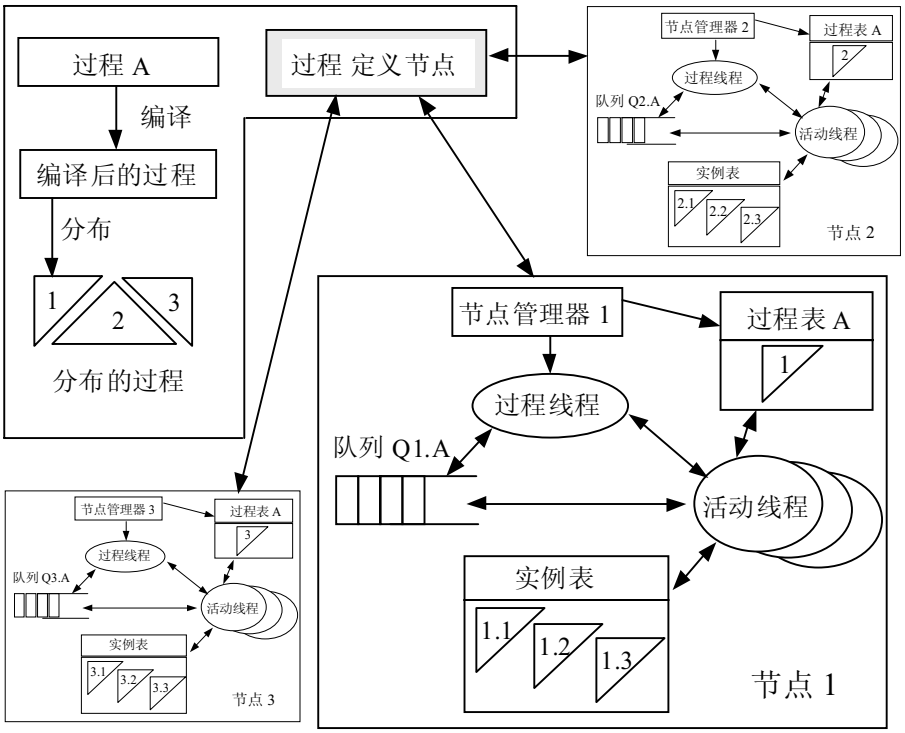


图 8.3 基于持久消息队列的系统运行示意

（3）工作流的执行过程

为了更清楚的说明工作流执行过程的原理，在此我们描述一个完整的工作流执行过程：

①一个工作流实例启动后，该过程实例中所有可以最先开始的活动实例（即只有输出控制连接弧而无输入控制连接弧的活动）将要被激活，这是通过向相应的执行节点发送消息，并把消息用 PUT 操作放入相应的消息队列中来完成的；

②执行节点上运行的过程线程不断地检查消息队列，一旦发现有新的消息，则从消息队列中浏览（而并非取出）它，然后生成相应的活动线程；

③活动线程从执行节点所维护的过程表中提取属于该活动实例的信息数据，利用这些数据，活动线程在实例表中添加有关该活动实例的记录；

④活动线程检验该活动实例的开始条件：如果该条件尚不能被判断（比如由于该活动的其他前趋活动尚未完成而使得相应的控制输入弧未发生转移），活动线程就在实例表中做上标记，然后即进入休眠状态，以等待过程线程在接到新的消息后再次激活它；如果该条件被判断为 False，则该活动被认为已经终止，并对相应的后继活动发出“无效路径削减”的消息，然后活动线程将实例表中对应于该活动实例的记录删除，并用 GET 操作将持久消息队列中所对应的消息取出，活动线程随即终止；如果该条件被判断为 True，活动线程则激活有关的应用，并从实例表中提取数据传递给应用以支持应用的运行，当应用运行完毕

时，再把应用的输出数据存放在实例表中；

⑤活动线程接下来开始判断终止条件：如果判断为 False，则再次激活应用；如果判断为 True，活动即被认为执行完毕，活动线程根据实例表中所记录的输出控制连接弧向其他节点发送消息，然后活动线程将实例表中对应于该活动实例的记录删除，并用 GET 操作将持久消息队列中所对应的消息取出，活动线程随即终止；

⑥以上这些对条件的判断及相应的操作，都是作为一个完整的事务来处理的；如果中间出现异常，则必须回滚至事务处理前的状态；

（4）讨论与评价

基于持久消息队列的分布式模型使得每一个单独的执行节点能够独立完成相应的工作流活动，从而实现了更为彻底的分布式的工作流运行方式，大大提高了工作流系统的可靠性；同时，通过增加具有工作能力的执行节点的数量，可以很容易地提高工作流系统的任务处理能力，增加其吞吐量，而避免了集中方式下服务器方很快出现的瓶颈现象。

但是，完全的分布执行也带来了一些麻烦，它使得在集中方式下很简单的一些问题变得复杂，比如判断工作流过程实例是否完全结束、工作表管理、对实例状态的监控、日志管理等等。因此，在具体的实现方面，还应该在这一基本模型的基础上进一步提出解决这些问题的方案。

尽管持久消息在系统出现异常的时候能够被完全恢复，但在某些情况下，全部使用持久消息所带来的开销可能会很大。系统设计人员应该根据实际情况，灵活地选择持久消息与非持久消息。比如工作流定义节点与其他节点间的联系就可以采用非持久消息，另外还有工作流监控与统计方面的消息等。

由于持久消息及队列是建立在底层的消息系统基础之上的，而不同的消息系统在具体的功能方面是有其自身的特点与局限的，因此，在这一模型的实现上采用不同的消息系统作支持将会面对不同的问题。

8.3.2 EVE

在多种工作流建模方法中，源于描述主动数据库（ADBMS）的 ECA 规则也被用来建立业务过程的工作流模型。ECA 规则定义了在某一事件下（Event），当满足定义好的条件时（Condition），被定义对象将执行的动作（Action）。相应地，在执行阶段，以事件为基础来构造工作流过程的执行模型也是一种可行的方法。

瑞士苏黎士大学计算机系的研究人员提出了一种基于事件的工作流执行服务中间件平台体系结构，称为 EVE（EVent Engine），用以集成工作流执行过程中松散耦合的分布式功能组件（包括各类企业应用）。

（1）代理

在 EVE 体系结构中，工作流的执行是由分布在网络上的代理（Broker）通过响应由 EVE 服务器检测到的事件来完成的；同时，代理在提供服务的过程中又会产生新的事件。每一个代理代表了一种活动任务的处理实体，它的行为也是由 ECA 规则来定义的。描述代理服务的 ECA 规则的基本形式如下：

ON service-request-event（service parameters）

[IF condition on parameters or broker state is true]

DO execute service provision actions;

generate reply event;

在 EVE 中定义了多种不同的代理，它们分别用于提供用户接口、组织管理、外部应用集成以及系统组件等功能。

（2）EVE 服务器

EVE 服务器是整个 EVE 系统的核心，它等同于 WfMC 参考模型中的工作流机。EVE 服务器能够直接同本地的代理及远程的 EVE 服务器相互通讯，而代理则只能通过 EVE 适配器（EVE-adapter）与本地的 EVE 服务器通讯。因此，不同代理之间的交互是通过把事件发送给本地 EVE 服务器，进而由本地服务器再发送给本地的相应代理或者再通过远程 EVE 服务器发送给远程的代理来完成的。

EVE 服务器控制着工作流实例的进程：它接收来自于本地代理及远程 EVE 服务器的事件，并根据运行库中的 ECA 规则定义，把事件发送给“感兴趣”的代理；同时，将这些事件记录进日志。当代理接到相应的事件后，就开始完成一个过程实例中的某一项活动，在这期间，它还会产生新的事件，比如 Action_***.done 等。这些事件被通知给 EVE 服务器后，EVE 服务器将继续以事件的方式推进整个过程实例的进程。

EVE 服务器的运行库保存了以下的重要信息：

①以 ECA 规则方式定义的不同类型的工作流过程；

②各个代理的能力表（包含代理所注册的事件）；

③代理的组织结构；

④各类事件的定义。

在系统实现上，EVE 系统的通讯基础结构是基于 ACE（Adaptive Communication Environment）环境建立的，EVE 服务器部分已经有了运行在操作系统 Solaris 2.5 上的软件系统，而 EVE 适配器则是依赖于客户端平台的，所以需要独立开发，目前，已经有了基于 Java 与 C++开发的两个 EVE 适配器软件。

（3）事件检测

每一个事件通常包括以下基本属性：

①事件类型；②事件发生的地点；③事件标志符（ID）；④事件的时间戳；

⑤所要求的服务名称；⑥事件发生所在的工作流实例的标志符（ID）；

⑦产生事件的代理的标志符；⑧事件的参数列表；……

事件的检测是通过 EVE 服务器内的事件检测器来完成的，包括对原语事件的检测以及对复合事件的检测。当代理通过 EVE 适配器向 EVE 服务器通告事件的发生后，EVE 服务器就把这一通告转给事件检测器进行处理，首先进行原语事件的检测，然后再进行复合事件的检测。

复合事件是多个原语事件的逻辑组合，可以用一个树状结构来表示，树结构的每一个结点都是一个事件类型（其中叶子结点必须是原语事件），而连接结点的边则代表了事件的组合。当某一结点的事件被检测到以后，即通知其父结点，看父结点是否能够发生：若可以发生，则继续向上一级的父结点推进；否则，就等待其他结点事件的发生以再次通知父结点。一个典型的复合事件如图 8.4 所示：

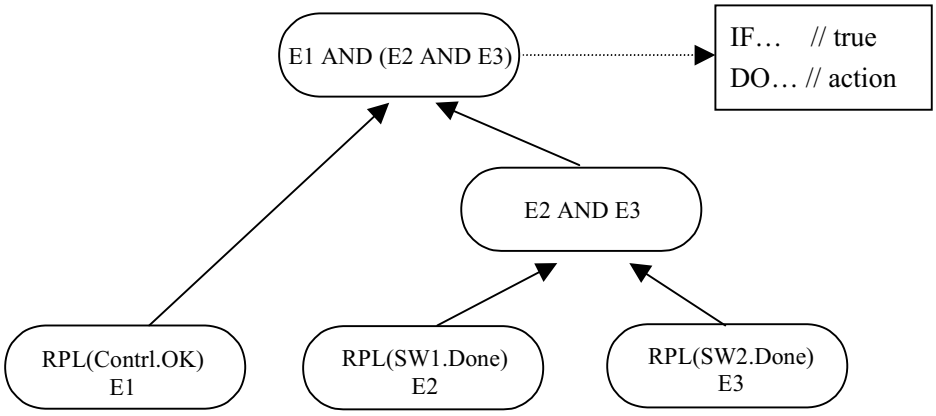


图 8.4 复合事件检测

（4）EVE 系统中工作流的执行过程

基于上述对系统各组件的讨论，给出如图 8.5 所示工作流的执行过程：代理产生事件→EVE 服务器接收→根据 ECA 规则通知提供服务的代理→代理执行服务并再次产生事件。

在图 8.5 中，由左至右按顺序给出了执行的四个阶段：事件发生、事件检测与记录、

规则执行、服务提供。这四个阶段是在分布的代理以及相应的 EVE 服务器的相互作用下而完成的，事件是驱动整个过程的源动力。

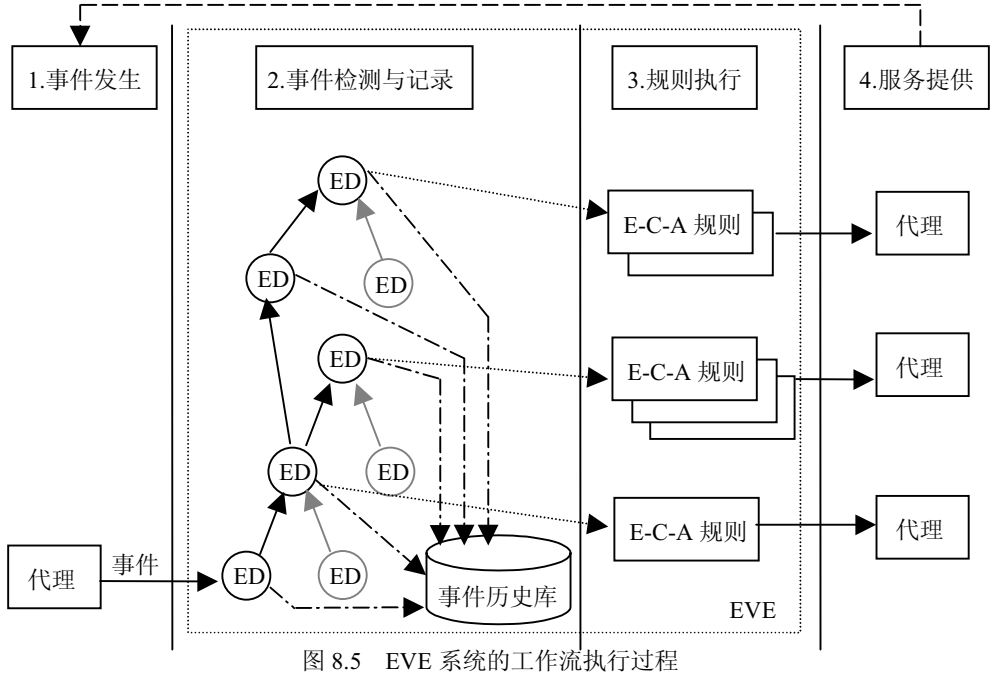


图 8.5 EVE 系统的工作流执行过程

（5）讨论与评价

基于事件的工作流系统具有如下优点：

①事件驱动为分布式系统提供了一种统一的组件行为描述机制，包括例外处理在内，都可以直接采用此种机制；

②通过严格定义事件语义，可以保证工作流的正确执行以及对它的监控；

③在工作流定义时不需要考虑完整的过程结构，以事件为核心的协调机制可以处理复杂的工作流过程；

④系统柔性增强，即使在工作流执行的过程中也可以修改过程结构；

从 EVE 系统的设计思路来看，它也属于一种完全分布的执行方式，因而很容易地克服了服务方完全集中于一点而带来的诸多不利，如系统吞吐量的瓶颈、系统的可靠性问题等。当然，也带来了一些复杂的问题，复合事件的检测就是一个例子。

8.3.3 DartFlow

DartFlow 是达特茅斯大学（Dartmouth College）计算机系开发的一种基于可移动代理的工作流系统。所谓“可移动代理”是一段可以在自身的控制下由异构网络系统中的一台机器转移到另外一台机器上运行的程序，也就是说，可移动代理能够在执行到某一点时挂

起自身程序，将代码传递到另外的网络节点上去继续运行。

可移动代理具有如下优点：

①减少网络流量：由于代理能够移动到网络上用户所需数据资源的所在地，省去了因中间数据传输所造成的多余网络流量，因而能够在一定程度上避免网络拥挤；

②适合于移动用户：代理的可移动能力使其特别适合于不具备可靠与持久网络连接的节点，如笔记本、膝上电脑等，即使最初生成移动代理的主机处于非连入状态，代理依旧能够自主地在网络上的不同机器间移动、执行；

③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；

④有利于数据集成：由于移动代理是到数据的所在地来访问数据，因此，数据源一方无需建立特定应用的远程数据接口，只需提供基本的查询接口，所有的中间调用过程都是在本地完成的；

⑤具有并行机制：移动代理是一段可执行程序，能够在主进程的控制下生成多个子进程同时执行，每一个子进程可以看成是一个子代理，再辅以一定的同步机制，并行执行将大大提高流程的效率；

DartFlow 是建立在可移动代理系统之上的工作流管理系统，它的组件结构与 WfMC 所提出的参考模型基本一致，能够相互对应，其中比较有特色的几个部分是过程代理、代理服务器、用户界面与工作表服务器。

（1）过程代理

过程代理实际上就是一个可移动代理，由于它自始至终地负责单独的一个过程实例的执行，所以在 DartFlow 中被称为过程代理。每一个过程代理拥有独一无二的一个标识号（ID），分别对应着独一无二的一个工作流过程实例。由于这种一一对应的关系，使得每一个过程代理能够很容易地做出改变来适应每一个实例的特定需要，独立地在系统中完成任务，这大大增强了工作流系统的柔性。

代理使用一种名为 TCL（Tool Command Language）的高级脚本语言，这是一种解释型语言，与 C、C++等标准的编程语言相比具有很多优越性。代理能够发送与接收 TCL 过程，一个正在运行中的代理可以重写它自身的过程甚至完全生成一个新过程，这使代理具有一定的“学习”能力。例如，当代理在执行过程中遇到例外情况时，它会与负责例外管理的服务器联系以获取解决例外的过程，而这些过程将会被写入该代理自身的过程当中，

当下一次遇到相同的例外发生时，代理便具有了自己处理这一例外的能力，无须再次与服务器联系。

过程代理通过执行任意复杂的过程来完成每一个任务（如访问数据库、访问打印机设备等），当它需要通知人来参与时，代理便移动到工作表服务器上，在那里将自身所携带的数据保存到磁盘上，并通知服务器在工作表中添加新的任务项，然后代理将自身也保存起来，以等待参与人员通过用户界面再次激活它。

（2）代理服务器

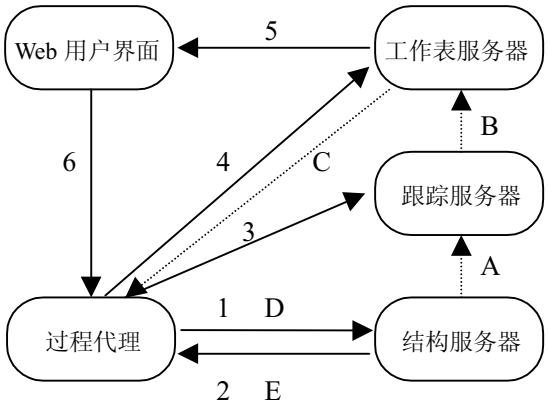
代理服务器实际上是为其他过程代理提供服务功能的代理，它保存了重要的过程定义和结构信息，是过程代理获取“知识”的源泉。在过程代理完成全部任务的过程当中，多个代理服务器将为它提供支持。我们将介绍比较重要的两类服务器：结构服务器（Organization Server）与跟踪服务器（Tracking Server）。

一个刚刚生成的过程代理不具备任何有关过程方面的知识，它所做的第一件事就是向结构服务器查询它所要完成的各项任务。结构服务器根据代理的过程类型以及它的来源向它返回一个具体的任务表，过程代理就根据这一任务表去执行实例。因此，结构服务器在 DartFlow 系统中实现了任务实例路由的功能。由于过程代理已经具备了执行过程所需的全部信息，在通常情况下（不发生异常、不进行动态修改），它不再需要与结构服务器进行交互。

跟踪服务器的主要任务是对所有已经生成的过程代理以及它们的活动顺序进行记录，而并非实时地跟踪。当一个新生成的代理从结构服务器那里获得任务表以后，它就要向跟踪服务器进行登记，一般是一个三元组信息 <代理标识号，过程类型，任务表> 。设置跟踪服务器主要是为了实现对过程实例的动态修改，以增强工作流系统的柔性。当需要修改某一实例中的活动时，首先通过结构服务器来获得修改的内容，然后由结构服务器通知跟踪服务器被影响到的实例，跟踪服务器根据自己所掌握的过程代理的信息，推断出受到影响的代理，并把这些信息发送给工作表服务器。当过程代理执行到某一活动需要生成任务项时，它就会移动到工作表服务器上，这样，工作表服务器就能够通知该代理返回到适当的结构服务器去修改过程信息，从而完成代理对动态修改的响应。由于跟踪服务器并不清楚某一时刻下某个代理的具体位置，但在工作表服务器的配合下，能够使过程代理在完成当前活动而返回工作表服务器时得到动态修改的通知，从而使这一动态修改能够在代理开始下一个活动时而生效。这种方案避免了因实时跟踪代理位置而造成的多余的网络流量，同时又能够比较及时地让代理对动态修改做出反应，不失为一种两全齐美的好方案。

当一个过程代理完成了过程实例中的全部任务后，它所做的最后一件事就是返回跟踪服务器并通知该服务器自己将不再处于运行状态，使跟踪服务器更新代理的状态信息。这个过程代理随后也就自动消亡。

图 8.6 分别给出了 DartFlow 中正常的过程执行顺序和动态修改的过程执行顺序，前者用数字表示，后者用字母表示。



正常过程：
1、 过程代理向结构服务器发送过程类型和来源 ID；
2、 结构服务器向过程代理返回任务表；
3、 过程代理向跟踪服务器登记；
4、 在每个任务结束之时，过程代理将通知工作表服务器；
5、 工作表服务器更新显示；
6、用户提交新数据，返回步骤 4。

动态修改过程：
A、 一旦有动态修改发生，结构服务器立刻通知受影响用户与过程的跟踪服务器；
B、 跟踪服务器通知受影响代理的工作表服务器；
C、 当下一次代理与工作表服务器交互时，工作表服务器检验该代理是否属于受影响之列；
D、 过程代理与结构服务器联系；
E、 结构服务器返回新的工作表。

图 8.6 DartFlow 中两类不同的过程执行顺序

（3）用户界面

DartFlow 的用户界面是基于 Web 浏览器的，通过嵌入 Java Applet（Java 小程序）的页面来与用户进行信息的交互。不同的 Java 小程序分别负责完成从工作表服务器上取出工作表、显示表单信息、保存用户修改后的表单等功能。基于 Web 的用户界面具有很多优点，如风格统一、简单易用、跨平台支持等，已经逐渐成为一种公认的发展趋势。

（4）工作表服务器

工作表服务器维护着工作流参与人员的任务信息，而且它还负责通知过程代理有关动态修改的消息。工作表服务器是过程代理生成任务项的唯一场所，因此，在整个系统中是不可缺少的。具体过程可参阅上文的叙述。

（5）讨论与评价

在 DartFlow 中使用了两种新技术——WWW 和可移动代理，比较好地解决了一般工作

流系统所欠缺的柔性、自适应性、规范性等问题。在系统中，多种类型的代理服务器构成了 DartFlow 的骨干，除了结构服务器与跟踪服务器以外，还可以加入例外处理服务器，以解决过程代理对付例外情况的问题，提高系统的可靠性。另外，增加同类型的分布的代理服务器，还有利于提高系统的整体性能，增强可扩展性。

第 9 章 CIMFlow 的系统设计

CIMFlow 是本书作者设计、开发的一个完整的工作流管理系统，它是根据当前工作流技术的主流发展趋势以及我国企业的实际需求而设计的。CIMFlow 提供了对企业经营过程建模阶段与运行阶段的全面支持，具有合理的分布式组件结构，系统中的组件包括建模工具、工作流机、管理工具、用户界面以及其他支撑组件（底层通讯基础、数据库等）。本章重点对 CIMFlow 的系统分析与设计过程进行阐述，以供有关的研究人员参考。

9.1 需求分析

目前，企业经营过程重组（BPR）已成为当今企业深化内部改革、提高竞争力的一个重要手段。但是，由于目前对企业经营过程重组的研究尚处于以概念、模型为主的框架性阶段，这使得人们在真正的重组实践中往往达不到预期设定的目标。BPR 总的应用状况并不乐观，有报告表明，70％的 BPR 项目未能达到预期的目标或归于失败。这一方面说明有关 BPR 的理论还不够成熟，另一个很重要的原因是人们在实施 BPR 的过程中缺乏有效的支持工具和评价标准，无法对重组的经营过程进行有效的仿真、定量的计算和分析，不能对重组后的结果进行合理、准确的预测。BPR 的实施过程具有很大的突变性，为企业带来很多危险因素，单纯的使用“头脑风暴法”、完全依靠人们的主观判断，是无法对一个复杂的系统进行成功重组的。工作流技术正好为减少这种重组的风险提供了一个非常合适的解决方案，为 BPR 的成功实施提供了有力的支持。因为先进的管理思想的运用如果没有科学的指导方法与成熟的辅助工具的支持，就无法有效地实现对重组过程的控制以及对企业各项指标的跟踪，也就不能在真正意义上实现企业通过不断完善、改进经营过程来不断增强适应市场的能力。支持企业经营过程重组的工作流管理系统的开发将为企业提供与此相关的软件工具支持，相应地，也必然有一套可操作的实施方法为企业提供指导。

从目前我们对国内外有关工作流产品的市场调查结果来看，在国外，工作流产品市场在整个 IT 应用市场上是较为活跃的一个部分，不仅开发厂商数量众多，从各个方面使企业用户有较为宽松的选择余地来挑选符合自己需要的产品；而且，企业应用的实际效果也较为成功，从而进一步刺激了企业实施工作流管理的热情，使工作流产品显示出良好的发展趋势，许多业界人士纷纷看好这个 20 世纪 90 年代才发展起来的新兴市场。在国内，由于

我国企业信息化进程远远落后于国外，在管理水平上也存在着较大的差距，因而，“工作流”对于我国企业而言还是比较陌生的一个词汇。这不仅表现在国内 IT 市场上自主开发的工作流产品的空白，同时，还表现在我国企业对国外工作流产品的引进与使用方面的欠缺。目前，国内的工作流产品市场尚处于一种萌芽阶段，可以预言，随着我国信息产业的飞速发展与国内企业的技术进步，我国的工作流产品市场必将是巨大的。因此，对工作流技术的研究是有着非常重要的意义的，CIMFlow 的设计与开发是为了适应我国企业对新兴 IT 技术的需要而开展的工作，希望我们在工作流管理系统方面开展的研究开发工作对于促进我国 IT 行业在流程管理软件方面的发展发挥一定的作用，为我国企业实施过程管理、项目管理、并行工程、供应链管理，进而实施先进制造战略能够提供方法上与工具上的支持。

在 CIMFlow 的设计开发过程中，我们充分利用了国内外已有的 BPM（经营过程建模）、BPR 以及工作流管理方面的研究成果（包括理论、方法和成功经验等），并在此基础上通过融合、消化及创新，形成了一整套具有自己特色的工作流模型与实施体系。我们要求 CIMFlow 具有较好的性能指标，友善的人机界面，高效的运行管理方式，能够适应企业的实际业务需求，具有可扩展性、易配置性，能够承担大吞吐量的任务。在研究过程中，我们自始至终贯彻这样的一种思想理念——“理论是 IT 发展的基础，IT 是为企业服务的工具”。理论的研究要充分考虑到它在 IT 领域的应用，要为 IT 的应用指明方向；而 IT 的实施则要以能为企业提高效益为关键，使企业能够因此而提高竞争力。作为理论应用于企业的中间媒介，IT 的作用是至关重要的。大量的案例研究表明，IT 是实施企业 BPR 的基础，没有 IT 就不可能将企业的组织运行机制同过程创新、跨功能的企业经营过程有效地集成起来，进而实现向面向过程的运作方式的转变。

9.2 CIMFlow 的工作流模型

工作流模型是整个工作流管理系统的基础，在进行工作流模型设计之前，我们首先制定了有下三个模型设计原则：

1） **面向企业用户，以简单、直观、容易掌握为前提**

在多种不同结构的过程模型中（如活动网络图、Petri 网、状态图、语义--行为模型等），活动网络图（Activity Network Diagram）是可读性最好的一种，对于非专业人员而言是最直观、最自然的过程表达方式，因此，CIMFlow 的工作流模型是建立在活动网络图的基础之上。

2） **过程描述能力强，能够定义可能发生的各种过程逻辑**

这一原则对模型的过程语义提出了较高的要求。模型在简单化的同时，必须要兼顾其自身的描述能力。除了常见的由 WfMC 定义的基本原语（如“与分支”、“或连接”等）外，过程中还可能出现更复杂的逻辑关系（如“A 活动的执行不能早于 B 活动”、“A 与 B 互斥，但其具体的选择机制则是依赖于外界的，是随机的”等等），这些情况也同样要求模型提供相应的概念予以支持。因此，我们需要在活动网络图的基本表达上进行合理的扩展，通过增加模型元素以及模型元素的属性来弥补其描述能力的不足，特别是要融入“状态”这一重要概念。

3） **应该体现企业这一复杂系统的多视图特性**

工作流模型应该是一种综合性的模型，不仅能够描述一个经营过程“是什么”的问题，而且还应该能描述“由谁做”、“怎么做”等方面的问题。因此，CIMFlow 的工作流模型在过程模型的基础上，需要加入描述企业人员的组织模型、描述企业资源的资源模型以及提供信息定义的工作流相关数据，以使其成为一个完整的、具有多视图特性的模型。

在以上这三条原则的约束下，所设计的 CIMFlow 工作流模型由四部分组成，这四个部分分别是过程模型、组织模型、资源模型以及工作流相关数据。

过程模型用来定义工作流的过程逻辑，它包括组成工作流的所有活动以及活动之间的依赖关系。它是整个工作流模型的基础与核心，其他模型均为其提供支持。在某些情况下，我们所说的“工作流模型”指的就是“过程模型”。

组织模型用来定义企业人员的组织结构，它包括几种不同形式的组织元素以及每种组织元素内部的递阶层次关系。组织模型的主要任务是为企业人员执行工作流提供柔性的组织定义，为过程模型提供“人”的支持。

资源模型用来定义企业资源的组织结构，它包括几种不同形式的资源容器元素以及容器内部的递阶层次关系，直至最终的原子级资源个体。资源模型的主要任务是为企业人员执行工作流提供“物”的支持。

工作流相关数据用来定义工作流执行过程中需要用到的数据，它包括简单的数据类型与复杂的企业对象，与 WfMC 所提出的工作流参考模型中的工作流相关数据具有相同的含义。它主要用于各种条件的判断，以实现工作流机对不同活动的选择性路由。工作流相关数据为工作流的执行提供了“信息”的支持。

整个工作流模型的组成结构以及内部各模型间的关系如图 9.1 所示。

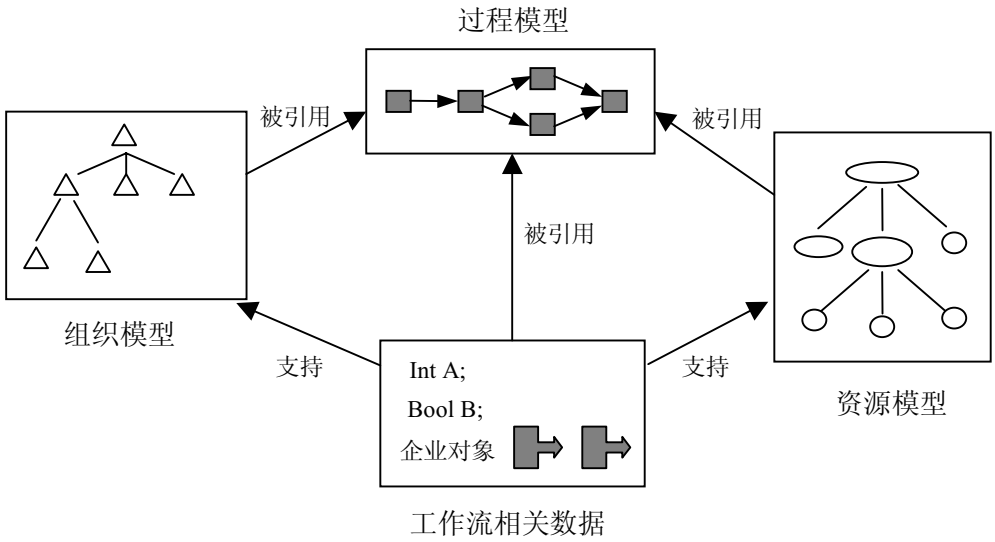


图 9.1 CIMFlow 工作流模型的组成结构

9.2.1 过程模型

过程模型的结构采用活动网络图的描述方式，一个工作流过程可以看成是一个由节点与连接弧所组成的有向图（允许自环的出现）。其中，节点代表活动，连接弧代表活动间的顺序关系。在活动网络图的基础上，我们又加入了“状态”与“条件”这两个概念以增强过程模型的语义，使其能够处理足够复杂的过程逻辑来满足企业用户的需求。下面就针对“节点”、“连接弧”、“状态”和“条件”这四个方面进行详细描述。

1. 节点

根据节点的功能，我们把节点分成三种类型，分别是任务节点、逻辑节点与标志节点。

1) 任务节点

任务节点代表了组成一个实际经营过程所需的各种类型的活动与任务。它包括原子级的“人工型活动”、“自动应用”以及非原子级的、可分解的“过程”。

(1) 人工型活动

“人工型活动”是指需要人的参与来完成的工作任务，通常是由工作流执行者从自己的工作项列表中来选择执行。当执行者完成任务以后，则通过客户端界面（Web 页面）对此活动进行提交，包括返回某些必要的活动处理结果等。工作流管理系统并不负责此类活动的执行，而是密切监视活动的状态，并管理由活动产生的工作流相关数据。在我国的企业中，特别是在自动化程度不高的企业或部门中，还大量存在着需要由人来完成的活动，

因此，CIMFlow 必须要提供对人工型活动的支持，包括定义与管理，特别是管理，因为对人工型活动实现有效地管理是提高企业运作效率的重要手段。

(2) 自动应用

“自动应用”是指不需要人的参与、直接由工作流管理系统激活相关应用来执行的活动。通常这一类活动体现了工作流管理系统与企业应用间的集成关系：二者间的集成度越高，则由工作流系统所管理的自动应用也就越丰富，企业的整体自动化水平也就越高。比较基本的自动应用包括打印服务、E-mail 发送、定时器、计数器等，其中定时器用来设定一个时刻或一段时间间隔，而计数器则用来对某些数据以用户设定的步长进行累加。比较高级的自动应用则可以是诸如 Notes、CAD 等应用系统，这些应用系统经过封装成一定的模块的方式集成到整个自动化的业务流程中。

从国外工作流产品的发展过程与未来趋势可以看出，应用的集成问题正在受到厂商们越来越多的关注。许多工作流厂商本身就是某一类应用（比如电子表格）的开发商，在该应用的基础上通过加入底层的通信基础结构（比如电子邮件系统、群件系统、CORBA 等）便实现了基本的工作流功能（即协作与集成）；还有一些厂商正好相反，他们是在本身具有的底层通信系统的基础上，通过构建上层应用来推出工作流产品。不论是哪一类型的厂商，它们一个共同的努力方向就是使自己的产品能够支持更多的操作系统平台、能够与更多的应用程序进行互操作、能够使用户以最简单的方法进行工作流应用的定制开发。总的看来，厂商们是通过以下一些技术来实现这些目标的：①跨操作系统平台的 Java 技术；②标准的异构系统互操作技术，如 CORBA、DCOM、MAPI 等；③API 接口；④可视化操作与 4GL 技术；⑤Web 技术；等等。

(3) 过程

作为组成工作流过程的最基本的原子级单元，“人工型活动”与“自动应用”是不能被进一步分解的。一旦我们的过程比较复杂，涉及的环节比较多，那么这些基本单元的数量也将大大增加，活动网络图中的节点数也必然会不断膨胀。这首先影响了用户对过程中各主要环节的把握和理解。事实上，我们可以把某些关系紧密的活动集合起来，在图上以一个节点表示，这就形成了“过程”的概念。

“过程”是一类能够分解的节点类型，它的内部可以包含组成工作流模型的所有元素，实质上就是一个子工作流。“过程”的引入增强了过程模型的表达能力，使模型具有了层次化的概念，并支持自顶向下的建模过程。在 CIMFlow 的模型定义中，“过程”可以出现在任意的模型层次上，即允许“过程”内部再次嵌入“过程”。通常，用户可以在模型的最顶

层全部用“过程”来表示，这样可以清晰的反映模型的总体逻辑结构，进而再在每个“过程”中详细地布置活动及其他模型元素，直到完成最底层的基本活动的建模。

任务节点的可重用性设计，统一任务节点的内部结构，建立输出→输入映射机制是提高建模速度和规范化建模活动的重要工作。由于任务节点对应了工作流过程中被实际执行的各类活动，因此它是构成工作流过程的最重要的部分，同时也是唯一需要讨论重用性的模型元素。用户通过建模工具建立了一个活动（也可以是“过程”）以后，这个活动应该被允许在多个不同的工作流中使用：从用户的角度来看，用户无须再重复“新建”过程，只要通过“引入”已有的工作流组成模块即可；从活动本身的角度来看，一个活动可能出现在不同的工作流过程当中，与其相关联的前趋活动与后继活动并非是一成不变的——这便是“重用”的含义。

被重用的模块可以覆盖模型的各个层次级别：包括活动（自动型与人工型）、过程（多个层次下的），甚至是整个工作流。建模工具应该向用户提供方便的、灵活的重用功能。在界面上，对“新建”与“重用”这两个不同的用户使用实例（Use Case）应有所区别。在重用时，用户是不具备修改原有模型的权力的。另外，系统应该提供相关的自动搜索机制，即在用户完成某一活动以后，能够主动提供可能的后继活动，这也是重用机制下的一个应用，即通过查找所有重用过该活动的工作流过程来得到可能的的后继活动。

我们先从基于组件与面向对象来开始考虑，对于那些可重用的组件与对象，都是严格定义了与外界的接口，并严格区分模块的内部属性与接口属性。在这种可重用的思想下，考虑任务节点的重用性问题，其关键在于统一任务节点的内部结构，从而建立输出→输入的映射机制。我们规定，任务节点的的统一的内部结构为 IPO 式，即输入→处理→输出。在这种 IPO 的结构下，节点间的连接是通过输入与输出实现的，因此，封装模块的内部细节，建立 O→I 映射机制，就能够实现重用。从宏观上看，可重用的工作流就是一个 IPO 结构；而经过分解细化后，内部的每个精细结构也同样是 IPO 式的。

这种 O→I 映射机制包括信息对象的映射（数据、文档等）与物理对象的映射（物料、产品等）。比如，对于自动应用，需要建立对输入命令行参数的映射，以使外部应用程序可以运行；对于人工型活动，也需要建立同样的映射，只不过映射的内容并非是命令行参数，而是具体的文档、物料等。在重用时，用户只需定义前趋活动的输出与后继活动所需的输入之间的对应关系即可，至于被重用模块的内部属性则无须重复定义

2）逻辑节点

与任务节点相比，逻辑节点并不代表真正需要执行的活动，它是为了表示任务节点之间的逻辑关系而设立的。在一个工作流过程中，活动间的逻辑关系并不仅仅是串行的顺序关系，还有可能出现较为复杂的“与”、“或”关系组合。在 WfMC 的标准文档中，就定义了六种基本的逻辑关系，它们是串行、与分支（并行）、与连接、或分支、或连接、循环。为支持逻辑关系的表达，在 CIMFlow 中引入了相应的逻辑节点来支持定义活动间复杂的逻辑关系。

在活动网络图的基础上，CIMFlow 中规定，在某一节点的所有输入连接弧（指向某节点的连接弧称为该节点的输入连接弧）中，只要有一条发生转移，则该节点即可执行；在某一节点执行完毕之后，则该节点的所有输出连接弧（由某节点发出的连接弧称为该节点的输出连接弧）都可以发生转移。这一规定在有向图中隐含了相应的逻辑关系，即不用特别引入其他元素，就表达了几种基本的逻辑关系，包括串行、或连接、与分支，如图 9.2 所示。

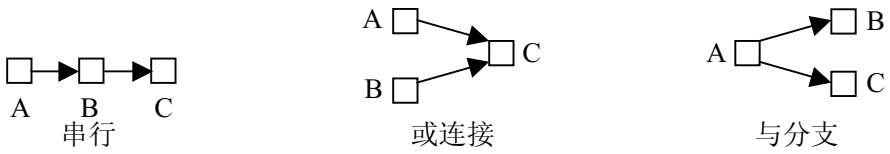


图 9.2 几种基本逻辑关系的表达

但是仅仅这几种逻辑关系是不能满足要求的，因此，我们又引入两种类型的逻辑节点来专门表达活动间的逻辑关系，它们是：“与节点”、“空任务”。

(1) 与节点

“与节点”是专门用来表达“与连接”关系的节点。虽然我们规定了活动网络图所具有的上述隐含的逻辑关系，但其中并未包括“与连接”的表达，因此我们引入了此种类型的逻辑节点。

“与节点”是一类具有特殊行为的节点：当它的一条输入连接弧发生转移后，“与节点”就开始执行，它的执行过程就是判断是否它的所有输入连接弧是否都已经发生转移。若全部输入连接弧都已发生了转移，则“与节点”执行完毕；若还有未发生转移的输入连接弧，则“与节点”仍处于执行状态，直到所有的输入连接弧全部发生转移为止。“与节点”的执行过程保证了“与连接”关系的实现，即只有当“与节点”的所有前趋节点都已经执行完毕后，“与节点”才继续激活后继节点。图 9.3 给出了“与连接”的关系。

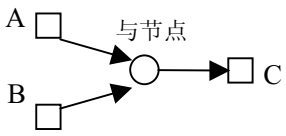


图 9.3 与连接

有一点需要指出的是，在有些工作流产品所提出的过程模型中（如 FlowMark、InConcert 等），并未把逻辑关系显式地表达出来，而是把它嵌入到了活动的属性当中，比如开始条件、结束条件等。尽管这种表达也能完成对逻辑关系的定义，但从图中并不能直观地反映出来，且容易产生混淆（如“与连接”和“或连接”），所以我们采用了显式的逻辑表达方式，即通过引入逻辑节点来直接定义逻辑关系。这种方式具有明显的直观性，可方便用户建模以及对模型的理解。

(2) 空任务

为显式地表达那些经过“与”、“或”组合而形成的复杂逻辑关系，我们引入了“空任务”这种逻辑节点。“空任务”就是没有任何执行过程的任务，一旦被激活，则立即结束，什么也不执行。它的唯一用途就在于提供了一种复杂逻辑关系的表达方式。如图 9.4 所示，节点 D 的执行必须满足（A∪B）∩C 的关系。

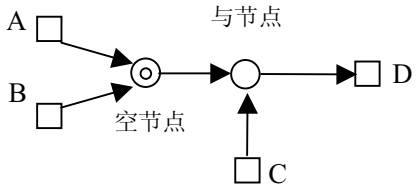


图 9.4 一个复杂逻辑关系的表达示例

逻辑节点的设立解决了显式表达活动间逻辑关系的问题，它使用户能够直观地理解流程的时序。但是，对于具有特殊要求的活动，这种显式的表达方式也无能为力。比如，活动 A 的执行时间不能早于活动 B 的执行时间，而 A 与 B 又并非是一般的串行关系。因此，过程模型还必须提供一种隐式的表达方式。我们采用基于状态和条件设置的表达方式，在下文中对“状态”与“条件”的描述中将会提到。

(3) 标志节点

标志节点有两种类型，分别是“开始节点”与“结束节点”。它们主要起到一种标记性的作用。由于活动网络本身是一个可能带有自环的有向图，而有向图是一种非线性的数据模型结构，与线性结构不同的是，它可能具有多个入口节点（即只有连接弧由该节点发出，

而没有连接弧指向该节点)；而且，在理论上选取图中的任一节点开始图的遍历都是被允许的。这就给用户与工作流机正确理解与解释流程的开始情况带来了不便，甚至会发生疏漏与错误。因此，需要引入一类具有特定含义的标志性节点——“开始节点”。我们规定，“开始节点”是一个工作流模型（或过程）的唯一入口点，它无前趋节点。一个流程开始执行时，最先被激活的就是“开始节点”，然后再由“开始节点”去激活后继的活动。

对于一个实际的经营过程，可能会由于不同的执行情况而出现不同的结果。对应于活动网络图，这种情况就表现为一次不会（也不可能）遍历图中的全部节点，只有部分路径被选择执行。相应地，模型图中就会出现多个出口节点（即只有连接弧指向该节点，而没有连接弧从该节点发出)，这些出口节点标志着流程的结束。为了清晰地表达流程的结束状态，并与“开始节点”相对应，我们引入了另外一类标志性节点——“结束节点”。类似地我们规定，“结束节点”是一个工作流模型（或过程）的唯一出口点，它无后继节点。一旦“结束节点”被激活，则标志着整个流程的结束。

标志节点的引入简化了对流程开始的定义与流程结束的判断。特别是对于分布式的工作流执行环境，这种简化将使工作流机无需运行多余的判别算法，只需按照模型定义来执行即可，流程的开始与结束都已经十分清楚了。

2. 连接弧

作为活动网络图中的另一类组成元素，连接弧是位于节点之间的有向线段，它从前趋节点指向后继节点。根据不同连接弧所表达含义的不同，我们把连接弧分成两大类：控制连接弧和数据连接弧。

1) 控制连接弧

我们规定，由控制连接弧所连接的两个节点，其行为规则是：只有当前趋节点执行完毕后，并且经过该控制连接弧的转移，后继节点才被允许执行。控制连接弧体现了过程的控制逻辑、节点间的时序关系，控制连接弧的转移意味着节点状态的转移与整个过程的演进。

控制连接弧发生转移是有条件的，因此我们为每一条控制连接弧都绑定了一个二值的布尔型转移函数 Trans(l) 。对于转移函数的组成，我们有如下定义：

转移函数是由一系列条件（Conditions）经过“与”、“或”组合而成的，其中每一个条件就是一个谓词逻辑，它的结果也是“真”、“假”二值的，这些逻辑的最终组合结果决定了转移函数的取值。

根据控制连接弧转移条件的特点，我们把在工作流模型中所应具体表达的控制连接弧类型分为如下两类：

(1) 永真型：即控制连接弧的转移函数值永远为“真”，Trans(l)≡1。这体现了一种顺序关系，不需要经过任何条件的判断，只要前趋节点执行完毕，即可激活后继节点。

(2) 不定型：即转移函数的取值是需要在具体的工作流执行过程当中由工作流机加以判断来确定的。这种判断实际上体现了一种选择关系，即根据不同的情况，通过满足条件的控制连接弧的转移，实现对某一节点的多个后继节点的选择性激活。

从直观和方便的角度出发，我们把“永真型”的连接弧称为“无条件连接弧”，而把“不定型”的连接弧称为“有条件连接弧”。实际上，“无条件连接弧”可以看成是“有条件连接弧”的一种特例，只不过前者的转移函数的取值恒为真。

通过“有条件连接弧”，我们可以表达“或分支”与“循环”这两种逻辑关系，如图 9.5 所示。对于“或分支”的情况，是当节点 A 执行完毕后，会根据执行结果来选择节点 B 或节点 C 继续执行，B 与 C 可能是互斥的；对于“反复”的情况，是当节点 A 执行完毕后，根据结果来决定是继续返回 A 执行还是执行后继节点 B。

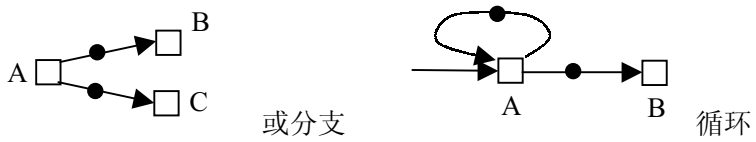


图 9.5 或分支与循环的表达

2) 数据连接弧

数据连接弧的引入是为了解决控制流与数据流不一致的问题。因为在有些情况下，某一节点执行完毕后，不仅仅要向经过控制连接弧与其相连接的后继节点提供数据，而且还有可能要向其他节点提供数据，而其他节点与这个执行完毕的节点间又不构成合理的控制逻辑上的顺序关系，无法用控制连接弧来表示。因此，通过引入数据连接弧，可以在单独存在数据关系的不同节点间建立连接，从而提供一种区别于控制连接弧的连接概念。

数据连接弧在它所连接的两个节点间建立了一种数据输出→数据输入的关系，前趋节点的输出数据可以通过数据连接弧来提供给后继节点，做为后继节点的输入数据。当然，数据连接弧不仅仅限于数据的传递，还可以包括物理对象的传递，比如在加工过程中生成的半成品、物料等。

综合以上对组成过程模型的各种元素的叙述，总结成图 9.6 所示。每一个不再被分解的叶子节点都将作为一个功能组件出现在建模工具的工具栏中，以提供给用户使用。

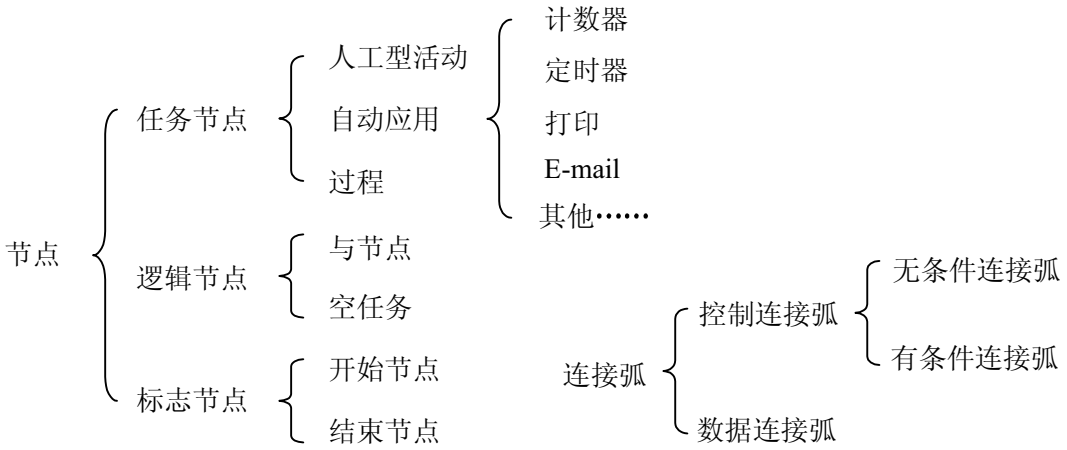


图 9.6 过程模型的主要组成元素

3. 状态

“状态”这一概念原本应属于工作流执行期间的概念范畴，在建模阶段就明确地提出它，主要是为了解决活动网络模型在状态表达方面的能力欠缺问题。与基于状态的过程模型（如 Petri 网）相比，活动网络图隐去了节点的部分可见状态，从而造成模型语义上的模糊，使过程模型的表达能力不足。举例来说，当某一控制连接弧发生转移以后，其指向的后继节点就被使能，但并不一定立刻开始执行，这种情况在活动网络图上就表达得很含糊。因此，在活动网络图中明确“状态”，建立一种与显式逻辑表达相辅助的隐式表达方式，将有助于提高 CIMFlow 适应复杂企业流程的能力。

对于一个可执行的活动，我们有如图 9.7 所示的状态转换图：

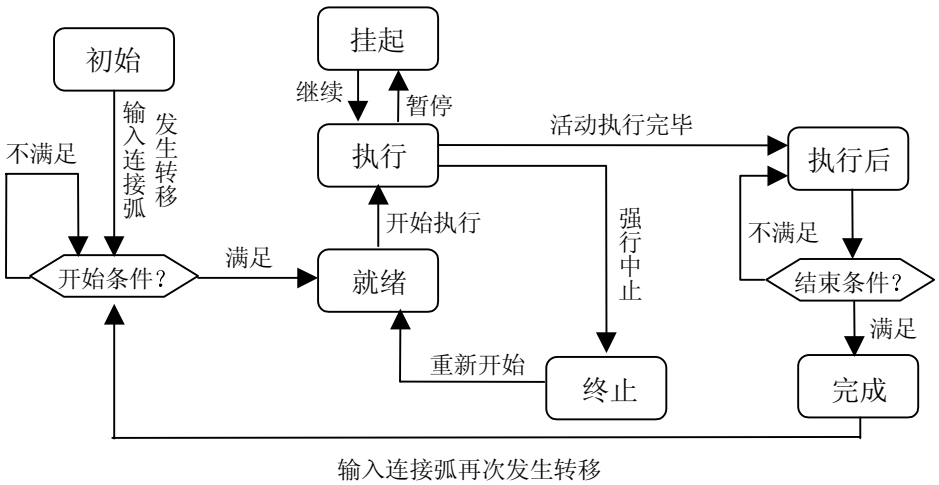


图 9.7 一个可执行活动的状态转换图

在整个工作流过程的控制流还未到达某一需要执行的活动时，该活动处于“初始（Inactive）”状态；对于已经执行完毕的过程，那些在过程中未被执行到的活动，将始终

处于“初始”状态。当活动的输入控制连接弧发生转移后，即开始检验活动内部所设定的“开始条件”，若满足条件，则活动将被使能，处于“就绪（Ready）”状态；若不满足条件，则继续进行判断，直到条件被满足为止。当被使能的活动经过触发机制（包括人工触发、自动触发）触发后，即由“就绪”状态转移到“执行（Executing）”状态。在活动执行的过程中，有可能发生一些例外情况，对于这些例外情况，可能使活动由“执行”状态转移到“挂起（Suspended）”状态或“终止（Terminated）”状态。对于“挂起”状态的活动，工作流管理系统将保留该活动实例的相关数据；而对于被“终止”的活动，则将完全清除该活动在执行时的相关数据。“挂起”和“终止”活动以后，如果可以继续或重新开始执行，则活动的状态将分别转移到“执行”状态和“就绪”状态。当活动经过一定的时间执行完毕以后，即进入了“执行后（Executed）”状态。我们之所以对“执行后”状态与“完成（Finished）”状态加以区别，是因为我们在活动中还设置了“结束条件”。若执行完的活动不满足“结束条件”，则不能转移到“完成”状态，这样在活动执行完毕的那一时刻起到满足“结束条件”这一段时间内，我们称活动处于“执行后”状态。设置“执行后”状态是有必要的，它可以对活动的完成加以精确地控制，而且有助于简化活动网络图的复杂程度。比如，活动 A 需要在活动 B 执行完毕后才能结束，在这种情况下，活动 A 就有可能处于“执行后”状态来等待活动 B 执行完毕。在活动处于“完成”状态后，如果需要反复执行，其输入连接弧则有可能再次发生转移，那么继续判断活动的“开始条件”，满足后活动将再次进入“就绪”状态。

总结上面对活动的状态转换图的描述，我们共设置了“初始”、“就绪”、“执行”、“挂起”、“终止”、“执行后”和“完成”七种状态，这些状态在建模阶段将隐含在活动的条件属性中，用于对过程逻辑的定义。由于从活动网络图中无法直接读出通过状态来控制的过程逻辑，所以我们称之为“隐式过程定义”。这种隐式过程定义将成为通过使用前面给出的模型元素来构造过程模型的显式过程定义的一种辅助手段，通过不同活动间的状态协同，对于处理那些较为复杂的、不太常见的过程逻辑将很有帮助，而且有利于简化活动网络图的复杂度。详见下节对“条件”的叙述。

4. 条件

条件是定义在工作流数据集（包括控制数据与相关数据）上的二值函数，用以决定过程中的活动在不同的情况下所要进行的不同处理方式。我们在 CIMFlow 的工作流模型中设置了三种类型的条件，分别是：转移条件、开始条件、结束条件。

转移条件是定义在控制连接弧上的条件，它是一种显式的条件定义，设有转移条件的控制连接弧我们在上文中称为“有条件连接弧”。转移条件决定了在工作流执行过程中所实际选择的由开始节点到结束节点之间的路径，不同的执行情况将造成工作流数据的不同，从而使不同的有条件连接弧发生转移来激活不同的后继节点。比如，对于两个互斥的活动，就可以通过设置完全相反的转移条件来保证二者只有一个能被执行。转移条件被满足后，相应的控制连接弧就发生转移，使后继节点能够按照状态转换图由“初始”状态开始进入不同的激活状态直至结束。

开始条件与结束条件是定义在活动内部的条件，它们是隐式的条件定义，分别决定了活动节点在什么情况下才能够真正开始执行和真正结束。在上面的状态转换图中，给出了这两种条件被判断后的状态走向：只有满足开始条件，活动节点才能由“初始”状态进入到“就绪”状态（即被使能的状态）；只有满足结束条件，活动节点才能由“执行后”状态进入到“完成”状态（即真正结束的状态）。开始条件与结束条件并不象转移条件那样能够决定过程的实际路径走向，它们只能决定活动的内部状态。在正常情况下，一个活动的输入控制连接弧发生转移后，它就一定要进入“执行”状态；而活动一旦执行完毕后，它就一定会进入“完成”状态。

在开始条件与结束条件的定义中，不可缺少的一个要素就是“状态”，这也是设置开始条件与结束条件的一个目的，就是把“状态”引入条件表达式中成为隐式的过程逻辑定义。用户可以把本活动节点的“就绪”状态、“完成”状态与其他节点的任意状态相关联，并加以时序上的限制（如“早于”、“晚于”等），从而实现活动执行过程中的同步。比如，活动 A 不能比活动 B 早执行，这种情况就可以设置开始条件：A.State.Ready<<B.State.Executing，而无需通过控制连接弧来进行定义；当然，控制连接弧也无法对这种情况进行定义，因为 A 与 B 并是那种直接的逻辑联系，仅仅是状态上的一种关联而已。

“状态”与“条件”虽然在很大程度上是图中一种不可见的隐式表达，但同样也是过程模型中不可缺少的一部分，它们使活动网络图具备了类似 Petri 网模型一样的状态表达能力，通过与“节点”和“连接弧”相互配合，可以使 CIMFlow 的工作流模型变得更为完善和实用。

以上四方面（“节点”、“连接弧”、“状态”和“条件”）的元素构成了 CIMFlow 工作流模型中的过程模型，它们将通过 CIMFlow 的工作流建模工具来提供给用户使用，以建立不同的工作流模型。“节点”和“连接弧”将以工具栏的形式出现，而“状态”和“条件”将以属性设置的形式出现。

9.2.2 组织模型

组织模型是用来定义企业中人的组织形式的模型，它应该提供灵活的结构以适应不同的企业或企业中的不同组织结构。在 WfMC 所定义的规范中，并未提供一个足够强大的组织模型，它只是对“工作流参与者（Workflow Participant）”在“角色（Role）”上加以区分，并由此建立了一个具有层次化结构的“角色模型（Role Model）”。显然，这种模型过于简单，很难适应企业复杂的组织结构。同样的，目前市场上的大多数工作流产品在组织定义这方面也显得能力不足，它们可以让用户定义任意的过程逻辑，但却不允许用户定义与他们的想法稍微不同的组织元素，这种不良局面使得企业在使用工作流管理系统进行运作时受到了很大的限制。

针对这一不足，许多研究机构（也包括一些工作流厂商）在组织模型方面展开了相应的研究，也提出了各具特色的组织模型（如 WIDE、MOBILE、DOPAS）。一个比较新的观点是“动态组织模式（Dynamic Organization Schema）”，它提出了两个基本概念：组织对象（Organizational Objects）和组织关系（Organizational Relationships），来允许用户根据自身的需要建立任意的组织形式。实际上，这两个概念也是通过事先定义好的元类型来提供的。考虑到大部分企业所共有的组织特点，我们认为：打破单一的“角色”定义，通过提供丰富的组织概念来构成组织模型是一种较为合适的方法。尽管这也属于一种预定义的组织模型，用户不具备自身定制的权力，但是用户仍可以通过这些预定义的组织模式来找到适合自己的一种组织形式，从而达到柔性建模的目的。

在 CIMFlow 中，组织模型由五种实体组成，分别是：“人员”、“角色”、“职务”、“部门”和“工作组”。它们的定义如下：

1) **人员**：对应于企业中每一个雇员，是一个独立的、具有一定行为能力和一定技术能力的人的实体；
2) **角色**：以技能为前提，能够完成某项功能的人员的总称，如车工、打字员、程序员；
3) **职务**：以行政责任为前提，代表了管理上的等级关系，如经理、科长、业务主管、职员；
4) **部门**：对应于企业的静态结构划分，这是由企业的实际部门设置情况来决定的，可以是传统的面向职能的，也可以是现在流行的面向过程与客户的；
5) **工作组**：以执行某一任务为目标而动态组建的、跨部门划分的一种组织结构。

组织模型中的这五种实体之间具有如下的几种关系：组成关系、负责关系、资格关系、

设置关系。图 9.8 给出了 CIMFlow 的组织模型图：

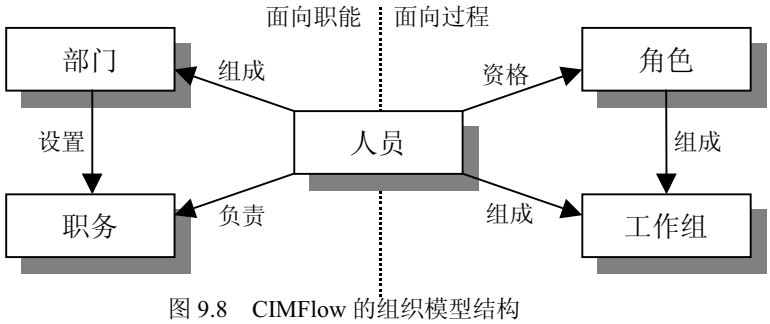


图 9.8 CIMFlow 的组织模型结构

组织模型中的五种实体分别体现了如下所述的不同的组织功能：

1) **部门**——结构上分解；

2) **职务**——上下级负责；

3) **角色**——功能上代替；

4) **工作组**——动态临时组织；

5) **人员**——基本组成要素。

“人员”直接对应于企业中实际存在的每一名员工，它是组成其他四类组织形式的最基本要素。每一个员工根据其掌握的技能可能具有一个或多个“角色”，而每一个“角色”下面也会对应着一名或多名企业人员。“部门”一般是由地理位置相同而且在企业内部具有相同任务目标的人员组成的，它体现了传统的面向职能的企业组织形式。在部门中根据责任的不同设置并划分了多个“职务”，职务间形成上下级的关系，下级应对上级负责。为了适应市场变化或者其它临时需要，不同部门、不同职务、不同角色的人员可以动态地组织起来，在一段时间内临时形成一种组织形式，就是“工作组”。工作组往往在完成了它的任务后就被解散，其组成人员也分别归于原来各自的部门。“部门”和“职务”是面向职能而设置的，“角色”和“工作组”则是面向过程而设立的。

在 CIMFlow 的组织模型中，除了“人员”以外，其他四种实体的内部结构都是一个自上而下逐层分解的树状结构，而每一个树状态结构的根节点就对应了这四种实体的名称。以“部门”为例，它的内部结构如图 9.9 所示。这种树状结构适合于所有企业的部门组织，只不过在节点与层次的设置上各有不同罢了。

组织模型的建立为工作流模型提供了有关人的视图，在用户通过工作流建模工具建立过程模型时，将会为每个活动节点设置有关组织的属性，即该活动由谁具体执行、由谁负责管理，这时过程模型将会引用到组织模型所提供的实体元素，用户也将会访问到某种实

体元素的树状结构。

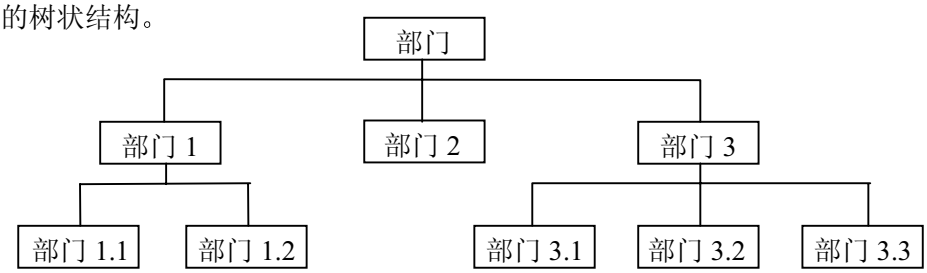


图 9.9 组织的树状分解结构

组织模型的数据建立与维护是通过专门的组织管理器来实现的，它是独立于工作流建模工具而存在的一个系统组件。对组织管理器的基本需求是——允许企业用户方便地建立自己的组织模型数据库，进行人员属性的定义与组织的划分，并提供方便、有效的管理维护界面，特别要保证数据的一致性，因为在组织模型中经常会出现一对多的实体间关系。

9.2.3 资源模型

资源是企业进行生产经营不可缺少的物的因素，在工作流的执行过程中，必须得到企业资源的支持，相应的组织实体才能够按照要求完成活动。从广义上讲，企业资源所涵盖的内容相当广泛，我们甚至可以说，企业中除了人以外的所有物质实体都可以称为资源。但是，在工作流概念模型中，不需要具体地考虑许多细枝末节的资源，而应该更注重在生产中起重要作用的那一部分，比较典型的资源如原材料、机床等。

在目前市场上的工作流产品中，几乎没有一种产品具有提供资源模型的功能，很大程度上的原因是这些产品往往不是应用于企业的生产领域。从工作流技术本身的起源来看，由于工作流是起源于办公自动化领域，因此更注重对文档信息的处理而并非是具体的生产制造过程，那么出现这种情况也就不奇怪了。一些研究机构（如 HP 实验室）虽然提出了资源模型、资源管理的概念，但也都较为简单，一般都是典型的树状结构。

尽管资源的种类繁多，但就其组织形式而言却没有人的组织模型那么复杂，因此，在资源模型中不必建立过多的实体元素，在 CIMFlow 中，仅仅引入“资源类型”与“资源个体”这两个概念用于资源模型的定义。

“资源个体”是原子级的、具体的资源对象，它不可再分，在过程模型中被整体直接引用，比如×××型号的车床。一些具有相同或相似功能的资源个体集合在一起就形成了“资源类型”，“资源类型”是可分解的，而且一般都分解为多个不同的“资源个体”，于是这二者之间形成了一种分解与包含的关系，如图 9.10 所示的资源模型结构图：

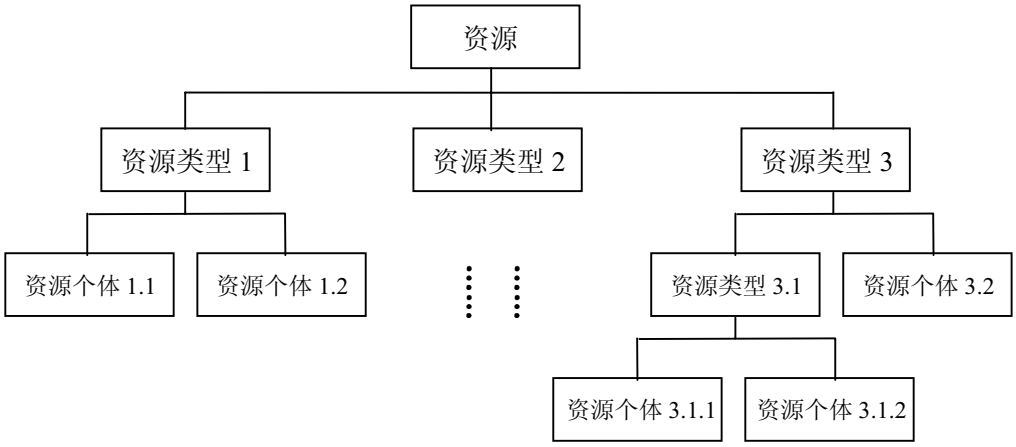


图 9.10 资源的树状结构图

在上图的树状结构中，“资源个体”都是叶子节点，而“资源类型”则都有父节点和子节点，其父节点是范围更大的资源类型（根节点为“资源”），其子节点是划分更细的资源类型或者是具体的资源个体。实际上，这样一个树状的资源模型图为企业提供了一个有条理的安排各类物质资源的组织方式，具体的资源个体都将最终被放置在树状结构的最底层，而资源类型的划分则为用户查找与定位一个资源个体提供了方便的途径。

同上面叙述的组织模型类似，有关资源模型的数据建立与维护问题，也是通过专门的系统组件——资源管理器来实现的。对资源管理器的用户需求与对组织管理器的需求基本相同，此处不再赘述。

9.2.4 工作流相关数据

在 WfMC 的规范文档中对工作流相关数据（Workflow Relevant Data）进行了定义：它是工作流管理系统在进行工作流实例的状态转换时所需要访问的数据，这些数据可以被工作流机、工作流参与人员和应用程序访问，并能够对其进行修改。一个完整的工作流模型必须提供给用户定义工作流相关数据的能力。目前，包含在 CIMFlow 工作流模型中的相关数据实际上是信息模型的一种简化，即仅仅提出了一些进行数据定义所需的实体类型，而并未涉及对实体间关系的讨论，因此，只能称之为“工作流相关数据”。

在 CIMFlow 的设计中，工作流相关数据主要包括两大类：简单变量和对象。

简单变量是对应一种特定数据类型的变量，包括常见的整型（Integer）、布尔型（Bool）、浮点型（Float）、字符串型（String）、日期型（Date）。从实用的角度出发，简单变量仅仅是单独的一个变量，并不提供对数组的支持。用户可以直接在工作流建模工具中进行简单

变量的定义，而且在定义时一般应给出变量的初始值或缺省值。

对象是一种封装了属性与方法的复杂的信息实体，实际上，被作为工作流相关数据的是对象的属性或者是方法函数的返回值。通常可以将企业中具有特定含义的信息实体做为对象，比如 Word 文档、工资表、提货单等等。在这些对象的内部隐藏了与具体的文件系统、数据库系统相交互的细节，用户只需了解对象属性与方法的含义，即可在过程模型中对其进行引用。有关对象的最初定义以及维护过程，是由专门的对象管理器来完成的，工作流建模工具只能对已经定义好的对象进行引用，自己并不能生成对象。

对工作流相关数据的定义是有作用域限制的，类似于全局数据与局部数据。用户可以对整个过程定义一个数据，那么这个数据就能够被过程中所有的活动所访问与修改，这就属于全局数据；用户还可以在一个活动内部定义一个数据，那么这个数据就只能由该活动的相关人员或者应用程序所访问并修改，对其他活动而言该数据则是不可见的，除非该数据通过控制连接弧或数据连接弧输出到了别的活动中，这就属于局部数据。

下面是一个工作流相关数据定义的例子：

RELEVANT DATA DEFINITION

NAME "document type"

TYPE STRING

DEFAULT_VALUE "Sales Order"

RESTRICTION "Available to all"

9.3 CIMFlow 的实现方案

CIMFlow 将以一套相互独立却又紧密集成的软件模块集的形式提供给最终用户，对应于建模阶段与运行阶段，分别给出了如下的系统组件：

建模阶段——工作流建模工具；资源管理器、组织管理器、对象管理器；一系列已封装好的应用程序模块（在建模工具中以工具栏的形式提供给用户）；一系列已定制好的企业对象（在对象管理器中体现）；一系列编辑好的 Web 用户界面；

运行阶段——工作流机（除了一组分布式运行的多个工作流机外；还包括一个完成总体控制功能的工作流机）、工作流运行管理器、工作表管理器/资源管理器、组织管理器、对象管理器，这几个模块在建模时与运行时提供的功能是不同的；

CIMFlow 的支撑系统包括数据库系统（SQL Server）、Web Server、CORBA 支撑环境。

由于 CORBA 及 Web 的多种优点，在 CIMFlow 中采用了 CORBA 结合 Web 的实现方式，以 CORBA 结合 Web 做为系统的底层支持，用 CORBA 来封装工作流系统的主要组件以及企业的原有应用，用 Web 做为客户端界面，用 Java 做为 CORBA 与 Web 的“粘合剂”。

图 9.11 给出了 CIMFlow 的系统结构示意。

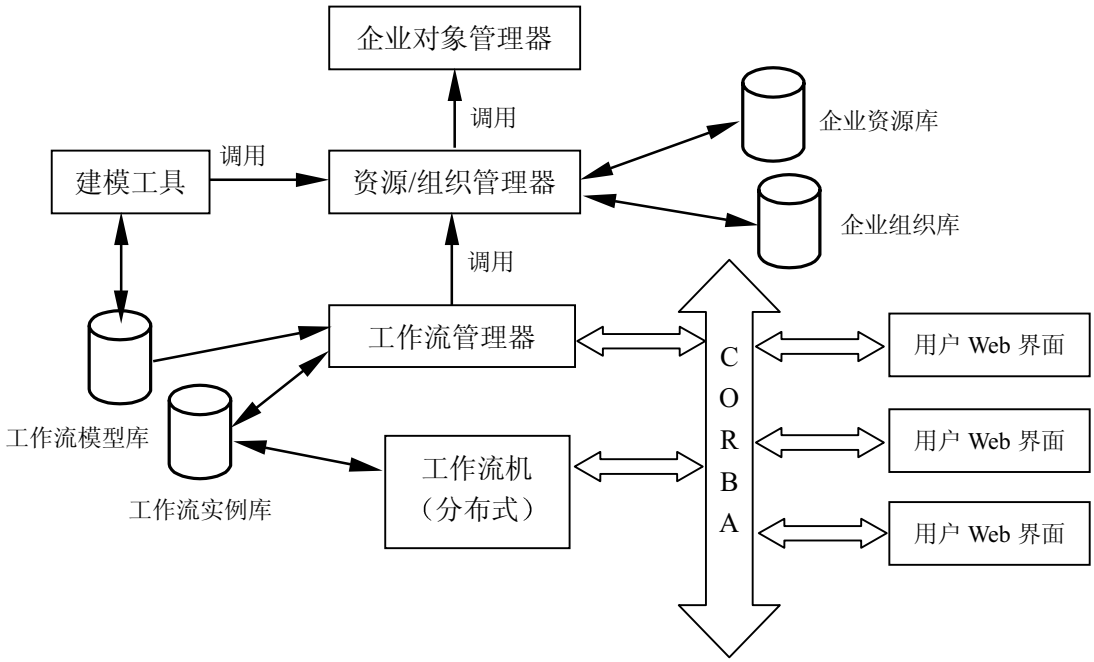


图 9.11 CIMFlow 的系统结构图

整个系统的所有组件都是以面向对象的方法设计、实现的，其中利用了面向对象的辅助设计工具 Rational Rose 来完成类的设计，用 Visual C++及 Java 语言来做为主要的系统实现语言。有关面向对象的辅助设计工具 Rational Rose 的详细介绍请参见作者的另外一专著《复杂系统的面向对象建模、分析与开发》[68]。CORBA 产品选择的是比较流行的 Iona 公司的 Orbix 及 OrbixWeb，它们分别提供了 IDL 到 C++以及 IDL 到 Java 的编译器。

9.3.1 CIMFlow 的运行过程

在图 9.11 所示的系统组件的支持下，本节介绍系统的运行过程，包括建模阶段与运行阶段两部分。

1．建模阶段

1） **建立过程模型：**用户利用建模工具，在可视化的界面下建立过程模型，即工作流的控制逻辑。

2） **建立资源与组织模型：**通过资源管理器所提供的资源视图、组织管理器所提供的组织

视图以及对象管理器所提供的企业对象的支持，用户可以在过程模型的基础上进一步引入各个活动所需的资源实体、组织实体及信息实体。

3) **活动执行角色定义：**对于人工型活动，建模人员需要绑定相应的用户界面模板；对于自动型活动，建模人员需要建立数据对象与命令行参数的映射。另外还需要考虑如何从应用程序得到应用执行的返回结果这个问题。在这个步骤完成工作流的执行逻辑的定义。

4) **工作流模型保存：**每个工作流模型将被保存在数据库中，这包括其中的各个活动、子过程、活动之间的控制流和数据流，以便于模块的重用。在工作流建模工具的主界面中，左侧是各个活动与子过程的树状列表（以唯一的 ID 号进行标识），右侧则是过程模型的视图显示。

5) **工作流模型分配：**用户通过建模工具把已经建立好的工作流模型分配给各个负责执行的工作流机，使每个工作流机在执行阶段都具有自主执行的能力，以充分发挥分布式工作流机的优点，解决集中式系统的运行瓶颈问题。

2. 运行阶段

1) **工作流实例的启动：**工作流实例有三种不同的启动方式——管理员启动（通过工作流管理器）、用户启动（由某些工作流实例的第一个活动的执行者来启动）、定时自动启动（由定时管理器完成）。

2) **工作流执行服务环境的建立：**工作流系统具有严格的执行顺序，（多个）工作流机之间也需要按照一定的逻辑来建立工作流执行服务环境。通常是先启动一个总控的工作流机，由它来同步所有其他工作流机，比如采用时间属性进行过程同步；然后再启动其他的执行工作流机（要向总控的工作流机注册，并由后者记录相应的配置信息）；与运行时相关的其他服务模块（对象管理器、定时器等）也要在有工作流实例投入运行前启动。

3) **工作流机实现的代码结构：**为支持多用户任务的同时响应，每一个工作流机从程序实现结构上来看都是多线程的，每个工作流机至少包括一个初始化的主进程，并能够根据响应的任务自动开辟新的线程，如图 9.12 所示。

4) **工作流机的执行过程：**每一个工作流机都被作为 CORBA 对象定义其 IDL 接口，工作流实例被启动后，通过调用 CreateProcessInstance 方法在相应的工作流机上产生服务线程。过程实例与子过程实例的执行过程都是相同的（即非原子级活动实例的执行），包

括初始化数据对象→修改过程控制数据（状态信息）→导航至可执行的原子活动→分配至某一工作流机执行（优先在本地开辟服务线程）→结束。

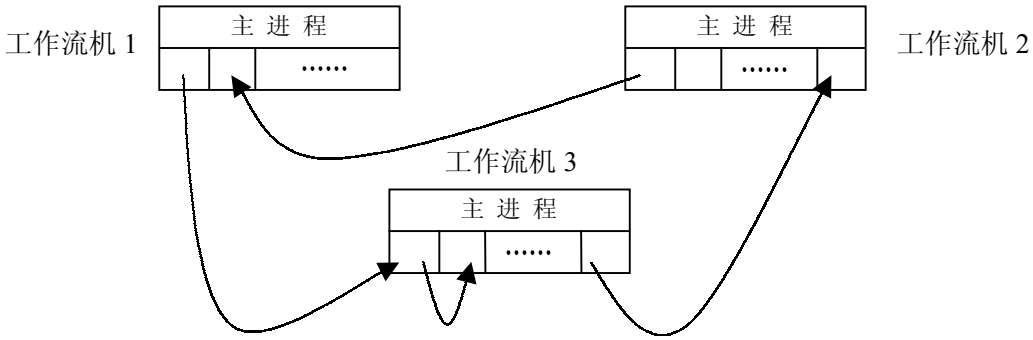


图 9.12 工作流机的内部代码结构

5) **原子级活动的操作：**对于原子级活动，有两种类型：人工型与自动型，相应的执行过程也分为两种：

- 初始化人工型活动的操作包括与资源管理器、组织管理器通讯，申请分配相应的资源与组织个体→与工作表管理器通讯，按照 PULL 原则（用户抢先式）为所有符合分配条件的人员生成任务项，并加入到各自的工作表当中→结束。
- 初始化自动型活动的操作包括激活相应的自动应用，传递参数，等待（或不等待）任务完成后返回。

6) **用户的参与：**工作流机在完成对人工型活动的初始化过程后，就暂时结束初始化线程，进入等待用户参与的阶段。当用户从任务表中真正开始执行某一任务时，工作流机将再次开辟服务线程来完成任务状态的转换、工作流相关数据的传递以及后继活动的导航等功能，所有的结果将通过用户的浏览界面予以显示，以便引导工作流参与者顺利完成任务的提交并开始另外的新任务。

9.3.2 工作流建模工具

CIMFlow 工作流建模工具提供的主要功能包括：

1) 提供用户建立工作流模型所需的工具栏；
2) 用户在 GUI 方式下对活动的属性进行编辑；
3) 用户在模型中定义工作流相关数据；
4) 提供过程模型与组织模型、资源模型的接口；
5) 实现模型文件的保存与读取；

6） 以树状结构显示工作流模型的不同视图。

图 9.13 给出了设计中的建模工具的主界面示意图，主要包括菜单、快捷键工具条、主窗口、视图窗口、建模工具栏、状态条。除了菜单与主窗口必须要显示出来以外，其他部分的窗口均可以由用户选择是否处于显示状态。主窗口是用户进行建模活动的主要区域，以图标的可视化方式显示用户所建立的工作流过程；视图窗口以树状的结构显示某一个视图，主要是显示过程视图，即显示当前建立的工作流模型中包含哪些子过程、活动和活动之间的连接关系，这样便于用户对所建的工作流模型的功能结构有比较全面的了解，也便于用户迅速查找和编辑某一活动。

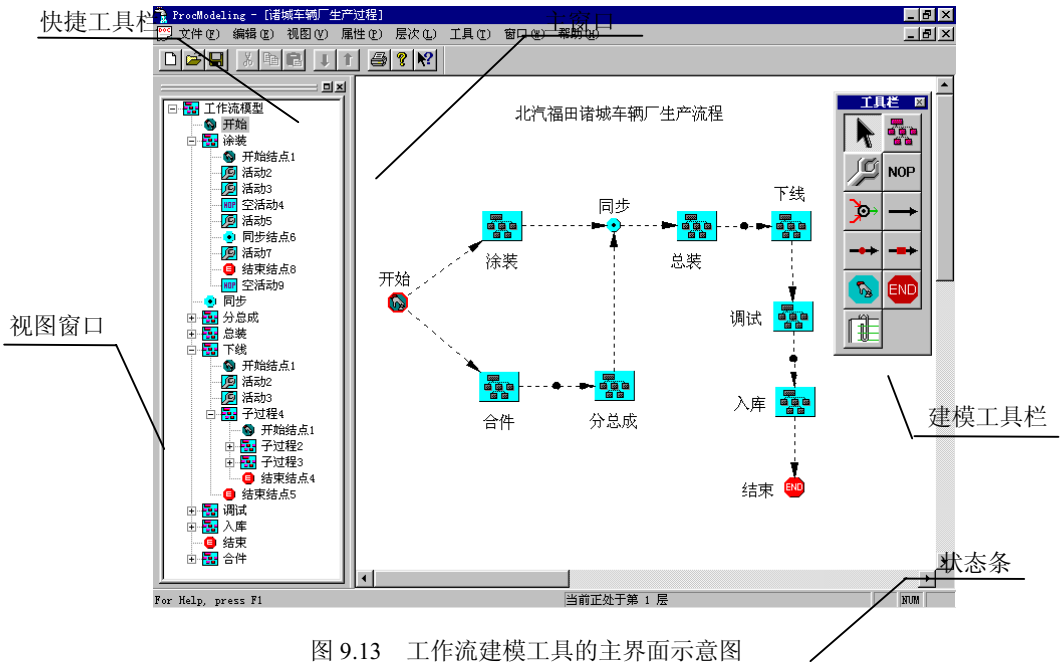


图 9.13 工作流建模工具的主界面示意图

建模工具栏为用户提供了建立工作流模型所需要的各类基本元素，根据 CIMFlow 的工作流模型，建模工具栏中包括：人工型活动、过程、与节点、空任务、开始节点、结束节点、无条件连接弧、有条件连接弧、数据连接弧、计数器、定时器这十一个工具按钮。用户只需要用鼠标从工具栏中选择相应的工具按钮，然后在主窗口当中单击鼠标左键，即可新建一个相应的模型元素；进而用户可以通过双击鼠标左键，在系统弹出的属性对话框中对该元素的有关属性进行设置与修改。这种可视化的操作使用户可以十分方便地建立自己的工作流模型。图 9.14 给出了人工型活动的属性对话框的主界面。

CIMFLow 完全采用面向对象的设计开发方法，对于建模工具提供的每一个模型元素，都有一个相应的类与其相对应。下面我们将给出利用 Rational Rose 这一 CASE 工具定义的 CIMFlow 工作流模型所对应的类图，以供读者参考。

图 9.15 给出了最顶层的类范畴图（每一个方框都表示了一个类范畴），每一个类范畴内部都包括了多个关系紧密的类。根据 CIMFlow 的工作流模型，相应地也就确定了四个类范畴。类范畴之间相互连接的一端带有小圆圈的连线表示“使用”关系，即过程模型中的类需要用到组织模型、资源模型以及相关数据中的有关类。

图 9.14 人工型活动的属性对话框示意图

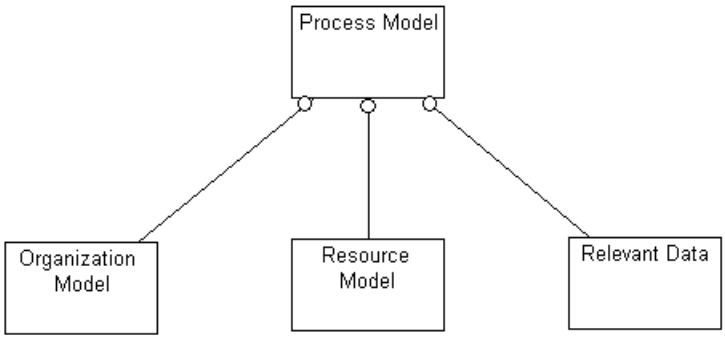


图 9.15 顶层的类范畴图

图 9.16 给出了过程模型所对应的类图。在图中，云状的图形表示一个类，类之间的连线表示类之间的关系。图中共用到了三种关系：一端带有箭头的连线表示继承关系；一端带有实心圆圈的连线表示聚合关系（包含关系）；一端带有空心圆圈的连线表示使用关系。从图中可以看出，WMObject 是一个最基本的基类，大部分过程模型中的类都是由此派生出来的；由 WMObject 派生出两个子类，分别是 WMNode 与 WMLink，分别对应于过程模型中的节点类型的元素与连接弧类型的元素；进而，由 WMNode 与 WMLink 再次派生多个子类，才生成具体的过程模型中的各个元素，包括 WMManualActivity（人工型活动）、

WMAutomatedActivity（自动型活动）、WMStartPoint（开始节点）、WMEndPoint（结束节点）、WMSyncPoint（同步节点）、WMNullActivity（空任务）、WMProcess（过程）以及 WMCondLink（有条件连接弧）、WMNonCondLink（无条件连接弧）、WMDataLink（数据连接弧）。WMNodeList 与 WMLinkList 分别构成了多个节点与多条连接弧的链表，WMProcess 与 WMNodeList 和 WMLinkList 是聚合关系（包含），即在 WMProcess 类中包含 WMNodeList 和 WMLinkList 这两个类，或者说 WMNodeList 和 WMLinkList 这两个类是 WMProcess 类的组成部分，因为一个过程需要保存其内部的节点与连接弧，所以这些节点与连接弧就构成了两个链表，即 WMNodeList 与 WMLinkList 类。类似地，在图 9.17、图 9.18 以及图 9.19 中，分别给出了组织模型、资源模型以及相关数据所对应的类图，类图中展示了构成这些模型的类以及类之间的关系。

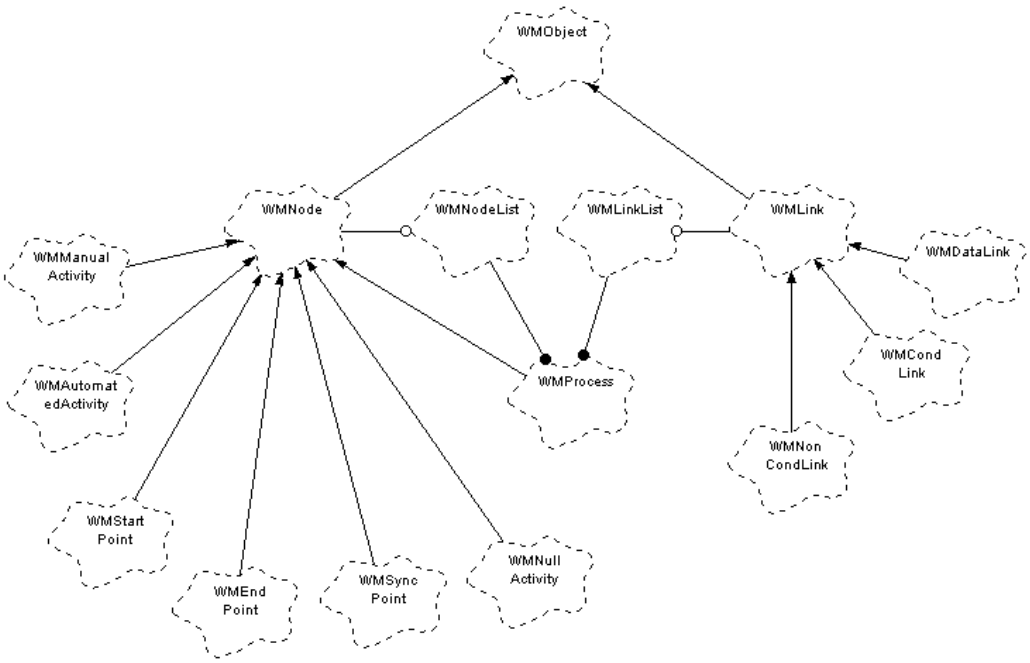


图 9.16　过程模型所对应的类图

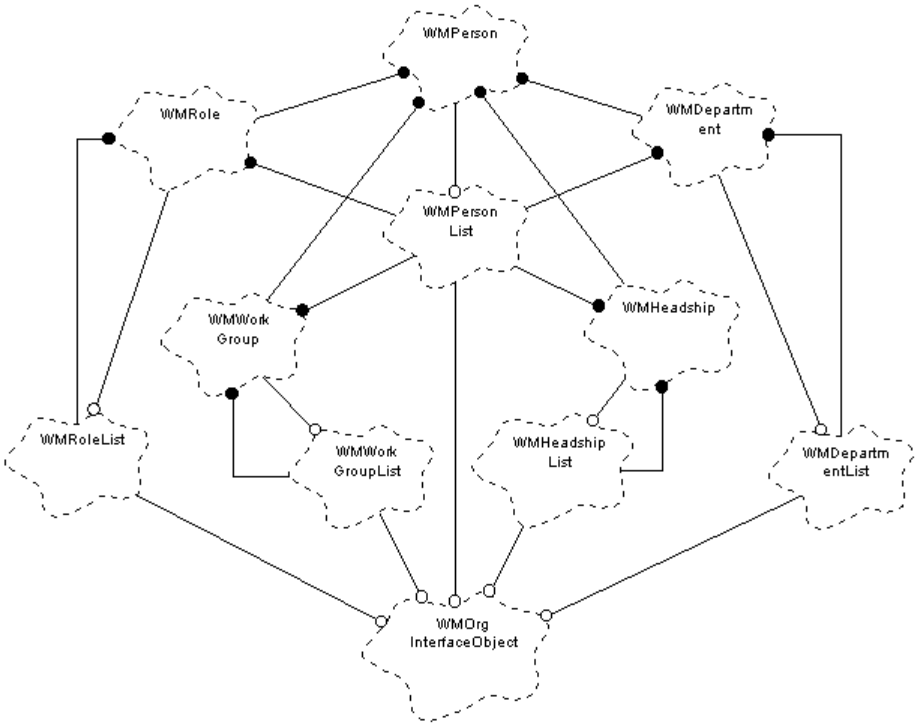


图 9.17 组织模型所对应的类图

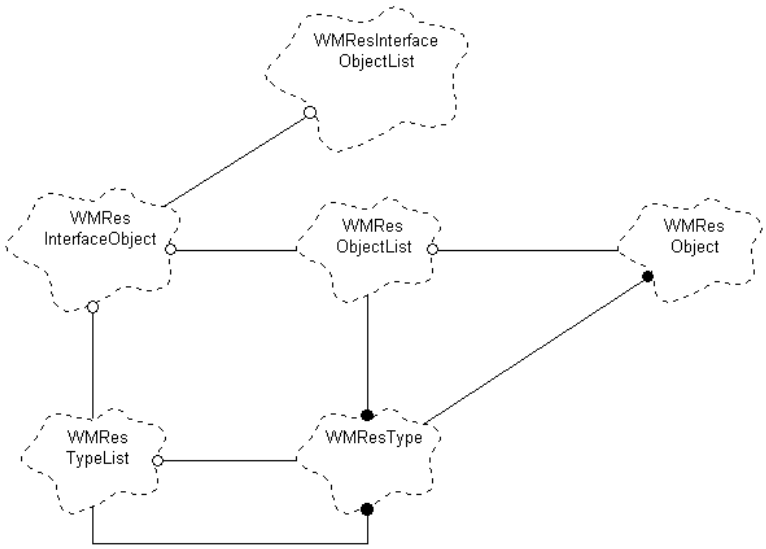


图 9.18 资源模型所对应的类图

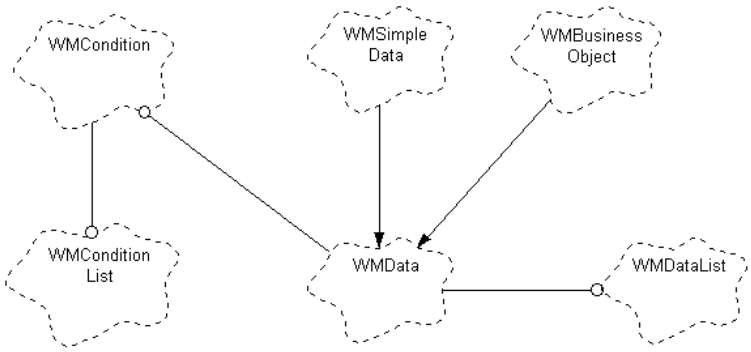


图 9.19 相关数据所对应的类图

9.3.3 分布式工作流机的设计方案

为了适应大规模企业业务处理的需求，CIMFlow采用了分布式工作流机的执行方案，其核心思想是：一个完整的工作流过程定义在由建模工具建立完毕后，对其进行合理地分割，然后再分别把分割得到的各个部分传递给相关的工作流机（我们称之为执行工作流机），使每个执行工作流机都保留一部分自己管辖范围内的工作流活动定义，这样，每个执行工作流机自身就具备了自主执行的能力。很明显，这种分布式的方案比传统的集中式的工作流机执行方案具有更好的柔性和更高的执行效率。

一个企业可以根据自身的实际情况，按照部门来配置执行工作流机，比如每一个部门配置一个，这种配置在建模时体现在对每一个基本活动的领域属性的定义上，建模时定义了不同的领域属性，在运行时就意味着将由不同的执行工作流机来负责管理、执行。

系统中设置了一个主控工作流机，记录各个执行工作流机的配置情况，并对执行工作流机的状态进行监控，同时负责生成工作流过程实例，并执行管理功能。

用户使用建模工具所定义的工作流过程将统一以数据库的形式保存在某一服务器上，这个模型可供管理员通过Web界面查询使用，主控工作流机运行在这一服务器上，处理这些查询，并管理这一核心的工作流定义数据库。主控工作流机与中央数据库位于同一节点上，使处理查询的过程本地化，有利于提高效率。

过程中的每一个活动以文件的形式被分别传送至相应的执行工作流机所在的节点上，保存在固定的目录下，供执行活动实例时引用。采用文件的方式将工作流活动定义发布到各个工作流机节点上有如下好处：①不必要求该节点一定安装有数据库；②处理速度快；③便于多版本管理，有利于活动的动态修改，增强系统柔性。

在由模型的集中式定义到分布式执行这一转变过程中，向各个执行工作流机传送活动

的执行信息是需要解决的一个关键技术问题。在集中式的建模工具中，活动与连接弧是相互独立的两种对象；而向执行工作流机传递时，则需要将这两类对象合并成为一种，即把连接弧的转移信息也加入到活动中来，使执行工作流机在生成某一活动实例后，能够直接确定所要激活的后继活动及相应的执行工作流机。因此，从分布执行的粒度来看，这种方案已经达到了原子级的活动水平，具有很强的柔性。

分布的执行工作流机得到了执行所需的过程定义后，就能够脱离开中央的工作流定义数据库而自主地按顺序执行，向下一个执行工作流机发出请求并生成后继的活动实例。而中央数据库由主控工作流机来管理与维护，它在执行期间不被其他工作流机调用。

我们以一个工作流的执行过程来介绍这种分布式工作流机的协作过程：

①根据不同的启动方式，通过不同的界面分别启动（包括管理员界面、对外的客户端Web页面、主控工作流机的管理程序等），启动命令将通过CORBA ORB调用主控工作流机的CreateProcessInstance方法来生成一个新的过程实例，该实例的相关信息以数据库形式保存，以便查询、控制；

②主控工作流机通过查询中央数据库以确定下一个活动由哪个执行工作流机负责执行，并通过ORB向该工作流机发请求，调用其CreateActivityInstance方法生成活动实例，并将该活动实例以文件的形式保存在该执行工作流机所在的节点上；

③主控工作流机继续调用执行工作流机的DisposeActivityInstance方法来处理这一活动实例：若是自动型活动，则由执行工作流机直接调用相应的应用程序；若是人工型活动，则向工作表管理器发请求，调用其CreateWorkItem方法来生成工作项（采用PULL型的工作表管理方式），

此时，主控工作流机的任务完成，以下的过程执行都是由其他各个工作流机来通过协作来自主完成了。

④当用户通过浏览器获取自己需要执行的活动后，点按“开始”按钮，则由嵌入到Web页面上的Java小程序来调用相应的执行工作流机的StartActivityInstance来改变该活动实例的状态，并由这些Java程序负责从执行工作流机上把用户所需要的数据信息传递到用户所在的机器上（如文件等）；另外，还需要通知工作表管理器对其他含有此项活动的用户的工作表进行清理，以免出现多个人执行一个活动的情况。

⑤当活动（包括人工型与自动型）执行完毕后，通过不同的方式（Web或者CORBA）来调用执行工作流机的CompleteActivityInstance，执行工作流机改变实例的状态，并进行相关数据的维护，然后根据活动定义来推进工作流进程，即调用负责下一个活动实例执行的

工作流机的CreateActivityInstance方法来生成活动实例。

图9.20给出了CIMFlow执行系统相应模块间的协作、调用顺序：

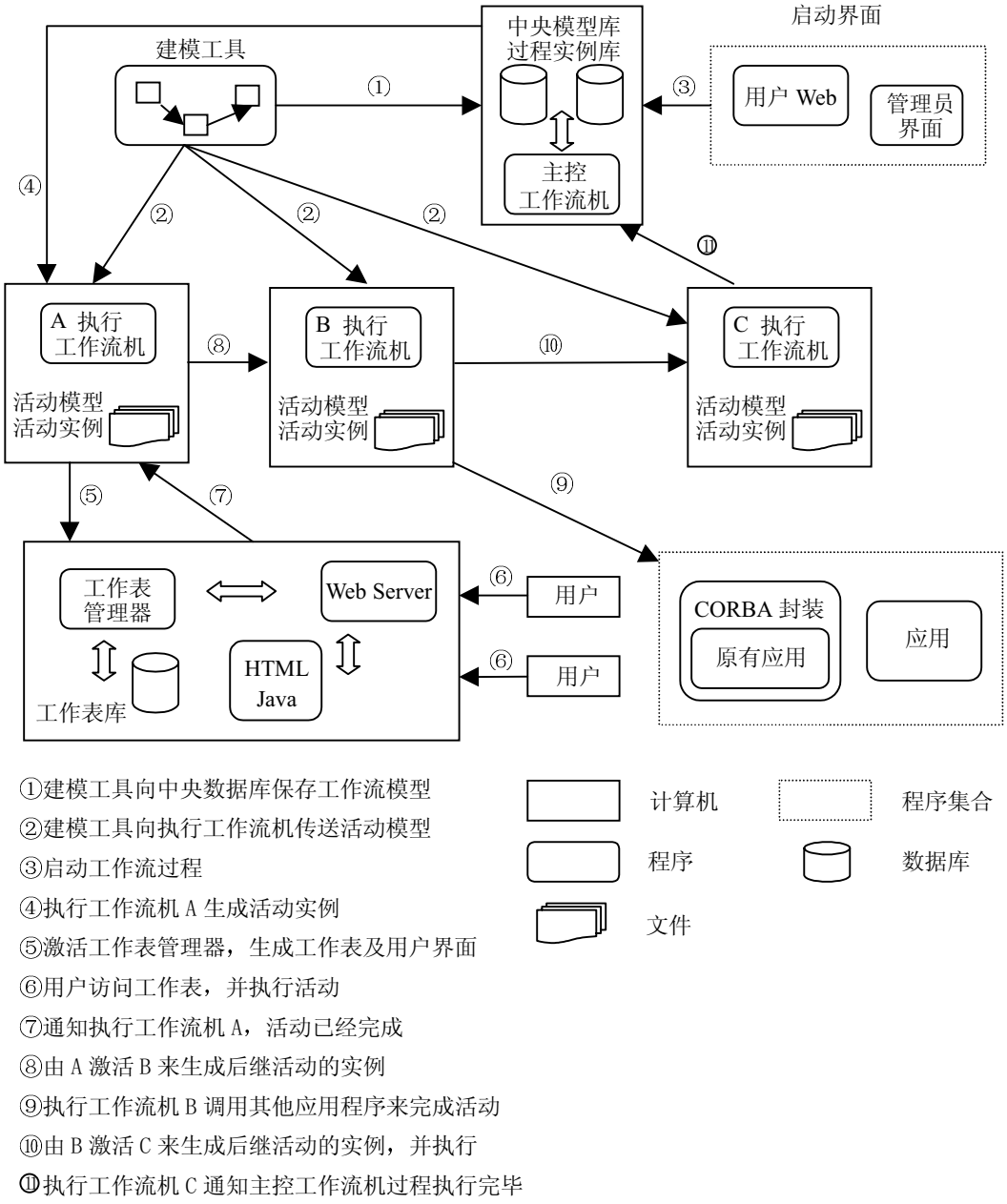


图 9.20 CIMFlow 各组件间的协作过程

对于一个已经预先建立好的工作流模型，有如下几种不同的启动方式：

①**外部客户启动：**由客户通过Web页面来启动一个工作流过程，比如网上订货。通常需要用户填写一些表单信息，然后通过点按“提交”按钮来开始；

②**由工作流过程中最先开始活动的执行者来启动全过程：**该执行者首先向工作流机登录，根据权限的设置，可以点按“开始新任务”（或类似）的按钮来启动；

③**管理员（或企业中有特殊权力的人）启动：**它们通过管理员界面或一般的用户界面直接

启动某一工作流过程；

④**系统自动启动**：对于一些例外处理、定期维护等事务，由系统根据消息或定时器激活自动启动；

9.3.4 工作流机的接口设计

工作流机是执行阶段的核心组件，它是以 CORBA 对象的形式来实现的，在本节中，通过对总控工作流机以及执行工作流机主要功能的分析，给出了主要的工作流机的 IDL 接口函数。这些接口函数首先以 IDL 文件的形式被定义出来，然后通过 Orbix 的 IDL 编译器映射成为真正的实现语言，在这里是被映射成为 C++。

首先来分析工作流机的几个主要功能：

对于总控工作流机，其主要的几个功能点如下——

（1）维护执行工作流机的配置，包括工作流机名、类型、IP 地址、启动时间等；

（2）将集中分布的工作流模型分布于各个执行工作流机上：这一功能是使总控工作流机作为工作流建模工具的服务端，将所建立的（或者从数据库中读出的）工作流模型分布到执行环境当中；

（3）监控执行状态：管理员通过 Web 对所有正在运行的工作流过程实例进行跟踪（查询状态）、控制（改变状态）；

对应于这些功能，设计如下的接口函数，在这里我们只给出接口函数名，对于具体的输入、输出参数，则予以省略。

（1）RegisterWfEngine（）：每个执行工作流机启动后，都要向总控工作流机注册，而后者据此更新配置文件；

GetCurrentConfig（）：执行工作流机从总控工作流机处获得最新的配置文件；

UpdateConfig（）：按照配置文件，新启动的工作流机或者将要关闭的工作流机通知其他工作流机更新各自的配置文件；

DestroyWfEngine（）：执行工作流机关闭，向总控工作流机取消注册；

（2）GetWfEngine（）：建模工具通过总控工作流机获取执行工作流机的地址、名称，以便进行模型分配；

DistributeWfActivity（）：建模工具通过总控工作流机进行模型分配，这一情况出现在当某一执行工作流机尚未启动，则模型信息将保存在总控工作流机处，等到该工作流机

启动后，再完成分配；

DistributeWfActivity（）：建模工具向执行工作流机进行模型分配，使执行工作流机获得可执行的活动信息，包括活动本身的内容以及相应的后继活动；

（3）GetProcessInstanceState（）：对过程实例的跟踪；

GetActivityInstanceState（）：对活动实例的跟踪；

SetProcessInstanceState（）：改变过程实例状态；

SetActivityInstanceState（）：改变活动实例状态；

CreateProcessInstance（）：生成一个新的过程实例；这一命令可以从管理员界面发出，也可以嵌入到最先启动过程的界面中发出。

以上五个接口实际上处理了所有从管理员界面发出的命令，包括启动某一过程实例、浏览实例信息、对实例进行紧急处理等。

（注：带有下划线的接口函数属于执行工作流机的 IDL 方法）

执行工作流机的几个功能点如下：

（1）被通知允许执行某一活动实例：

（2）生成任务表（这是一个内部功能，对其他组件而言是不可见的，因此无需设计其 IDL 接口函数）：按照分配原则针对不同的人员生成他们的任务表；

（3）用户开始、结束某一活动实例：用户从 Web 界面上点按“开始”、“结束”等按钮，对活动实例的状态加以改变；

（4）对数据的管理：

主要的接口函数如下：

（1）Run（）：前面的执行工作流机首先根据当前实例的信息获得后面工作流机的引用，然后调用该方法将过程导航到下一个工作流机处，工作流机将根据活动实例生成多个任务项或者直接激活某个应用，同时还将对全局实例库中该过程实例、活动实例的状态进行更新，另外还将在本地生成一个基于文件的实例存储，将主要记录实例的数据。

（2）StartActivityInstance（）：用户开始某一任务，执行工作流机将负责把与活动有关的数据传递给该用户（或者直接启动某一应用）；用户对工作流机的引用是通过实例数据库中的有关记录获得的，因为在启动这一任务之前用户至少已经浏览了自己的任务表，相关的信息就可以在那一次对数据库的操作中提取出来。

(3) FinishActivityInstance（）：通知执行工作流机某一任务已经完成，该工作流机便开始生成相应的 Web 界面来从用户那里获取相关的输出。

(4) SubmitActivityInstance（）：向工作流机提交输出项。

（5）GetAttribData（）：从某一执行工作流机的实例文件当中取得属性值，比如变量值等。

作为整个工作流系统的支撑组件之一，数据库的使用是必不可少的，它负责保存模型、实例以及其他方面的重要信息。通常情况下，需要设计如下一些数据库：

(1) 全局模型库——保存所有已定义好的工作流模型；

(2) 全局人员库——保存所有的工作流参与者信息，包括其所属角色、用户名、密码等；

(3) 全局实例库——保存所有执行（包括已经执行完的）过程实例、活动实例的信息，在一定程度上类似于日志文件，对查询提供支持，实例库中将只保留控制流方面的信息，而有关数据则保存在各个执行工作流机处的实例文件当中。

(4) 全局任务库——保存所有人员的任务表；任务表中要保存相关的详细信息，以使用户在改变实例状态之前所有操作都无需与工作流机交互，而是直接提取数据库的信息。

9.4 CIMFlow 的 Web 界面

随着世界范围内 Web 技术和电子商务应用的飞速发展和普及，采用浏览器/服务器（B/S）模式的软件系统已成为当前应用软件系统发展的趋势。对 Web 技术的支持给用户使用带来了很多方便，用户无须在客户端安装专门的软件就可以调用工作流服务器端的功能，交互界面风格统一，易于使用，特别适合于跨平台的分布环境，将来，基于 web 的工作流技术将进一步发展，工作流可以通过 web 扩展到多个企业，而且多个 web/工作流服务器的协同工作也将成为可能。

根据系统的实施方案，CIMFlow 将支持用户使用 Web 浏览器来实现工作流管理系统在客户方的功能：允许用户从 Web 浏览器中启动和控制一个工作流实例的运行，支持通过 Web 来管理任务项列表。按照界面所能实现的功能来划分，CIMFlow 的 Web 界面可以分为管理员界面与普通用户界面两个部分。

9.4.1 管理员界面

管理员界面的功能是完成对系统的监控和管理。监控和管理的功能包括很多方面：从这个界面上，管理员可以进行系统的登录与注销，查看当前系统的工作流机配置与运行情况，了解过程与活动实例的各种属性，并能够对过程与活动的执行进行人为干涉与改变。

除此之外，管理员通过 Web 页面实现对组织人员的管理，也是一个很实用的功能。所设计的几个主要功能界面如下：

1） **管理员登录和退出界面**

管理员的登录界面是为系统的安全性考虑设置的身份验证界面，通过管理员 ID 和口令实现对访问权限的限制。通过身份验证后，建立同工作流机的连接，允许管理员进入管理界面，进行后续操作。管理员工作完毕退出系统的同时进行注销的操作。图 9.21 给出了管理员登录和退出的界面。

图 9.21 管理员登录与退出界面

2） **查看系统配置界面**

由于 CIMFlow 系统是分布式的，系统中的各个工作流机动态的分布于不同位置，因此管理员需要了解系统中所有正在运行的工作流机的配置情况，包括工作流机的编号、名称、所属领域、当前所在的 IP 地址、当前状态等。查看系统配置界面使系统管理员能够方便地获取这些工作流机的相关配置信息。图 9.22 给出了查看系统配置的界面。

图 9.22 查看系统配置界面

3） **查看过程与活动实例**

查看过程与活动实例的界面是管理员对过程实例和活动实例的执行情况进行监控的界

面，主要用来显示过程实例与活动实例的相关状态信息，以便执行管理员的控制命令。这部分界面能够显示过程模型的拓朴结构，能够以不同颜色来明显区分过程实例中不同活动所处的不同状态，能够显示并修改活动实例的属性，如图 9.23 所示。

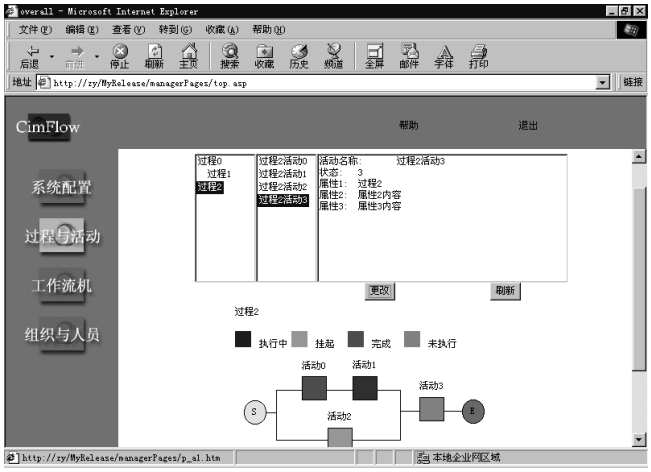


图 9.23 查看过程与活动实例的界面

4) **工作流机的工作负荷监视界面**

工作流机分布在不同位置的不同计算机上，为了协调和平衡各个工作流机的负荷，管理员应可以查看比较各个工作流机的负荷情况，即每个在线工作流机上执行哪些活动以及已完成和未完成的活动的数量（这里以活动数为统计单位）。这些内容在这里以统计图表和活动列表的形式显示，直观明确、一目了然，并可以此为依据进行定期统计，生成报表，便于优化。图 9.24 给出了工作流机的负荷监视界面。

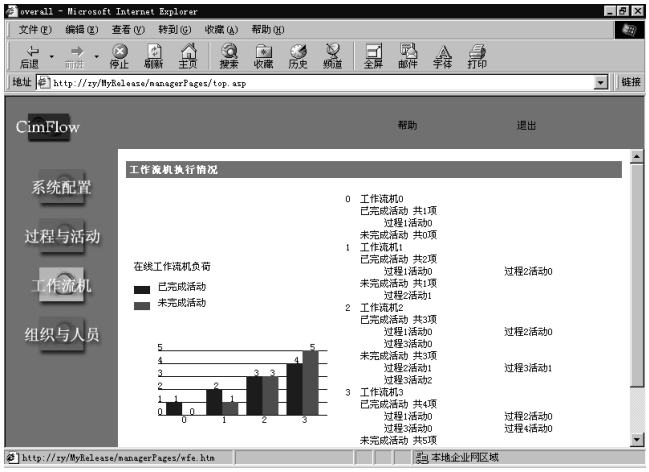


图 9.24 工作流机的负荷监视界面

5) **组织人员信息管理界面**

组织人员信息为管理员查询浏览以及重新分配角色或执行人员提供了依据，这里列表显示了人员的详细信息，包括部门、职务、角色、联系方法等等，供管理员查看。

6） **帮助信息界面**

提供关于界面使用方法的信息。

9.4.2 普通用户界面

普通用户的界面，主要是针对负责执行人工型活动的大量工作流执行者而设计的，其主要功能为查询工作表、确定开始某一项合适的任务项、提交完成任务项以及挂起、中断等等。图 9.25 给出了其界面形式。

图 9.25 普通用户的任务项列表界面

用户界面由框架分为三部分，顶部是系统封面图片，左边是功能菜单，包括“任务列表”、“查找”、“帮助”、“退出”四项，采用超链接，选中某项后，在右侧主窗口中装入相应页面。整个用户界面形式类似于电子信箱，可以使用户感到熟悉易用。

中间的任务表分别列出了每条任务的 ID、名称、摘要、完成期限、开始时间、结束时间和状态。每条任务前设置复选框，用户选择一条或多条记录后，点击“详细显示”按钮，提交表单，即转到详细显示页面。每条任务的摘要栏也链接到详细显示页面，点击后，在弹出的窗口内显示该任务的详细内容，如图 9.26 所示。

图 9.26 任务项的详细显示

任务表最右端的“打开”图标链接到任务操作页面，点击某条任务的“打开”图标，该任务的 ID 即作为参数传到操作页面，在那里可改变这项任务的状态。图 9.27 即为用户的任务操作界面。

图 9.27 用户任务操作界面

在界面的左侧列出了所选任务的简要信息，中间一列的上方显示任务的内容和附件，附件同样可以打开；下面则是输入输出文本框，供用户在对任务进行操作时输入或读出需要工作流机接收或给出的相关信息。右侧是对任务进行操作的按钮和操作状态显示窗口，可以进行的操作有“开始”、“完成”、“挂起”、“继续”，点击相应的操作按钮后，用户就开始与工作流机进行交互，输入数据将被传递给工作流机，并由工作流机判断是否允许请求的操作（例如只有在开始之后才允许完成的操作），然后做出应答并把结果传回用户界面。

第 10 章 工作流技术在企业经营过程重组中的应用

10.1 企业经营过程重组的概念

10.1.1 企业经营过程重组的基本概念

企业的经营过程是指企业为了实现一定的经营目的而执行的一系列逻辑相关的活动的集合[46]。经营过程的输出可以是能够满足顾客或市场需要的产品，也可以是特定的服务。

虽然经营过程的概念是近年来才被明确提出的，但自企业诞生之日起，经济学家和企业家们就一直在探索能够有效地管理企业中各项活动的方法和途径，以追求更大的企业效益。1776 年，英国经济学家亚当·史密 在其名著《国富论》中提出了劳动分工的原则。根据劳动分工的原则，每个生产者从事的生产操作相对固定，从而提高劳动者的劳动技能和熟练程度，减少因工作变换而损失的时间。在“分工理论”产生 100 多年后，泰勒、法约尔、福特和斯隆等人进一步地发展和丰富了分工理论。泰勒提出以标准化、系统化和科学化的管理代替过去的经验管理； 法约尔对管理本身进行了明确合理的分工，使管理从生产中独立出来；福特在汽车装配工序中将分工进一步细化，经过多次工艺改进和层层紧密分工，终于创造出了“流水线生产方式”，使效率得到大幅度的提高。斯隆创立了“分权管理”的管理体制，对管理职能进行了分工。这种分权的管理体制至今仍是各大型公司的主导模式。

分工理论的形成与当时生产力水平低下、商品供不应求的社会经济状况密不可分。无论是亚当·史密斯的分工理论，还是后人对分工理论的进一步发展和应用实践，其最终目的都是为了提高劳动生产效率，大批量生产是当时企业的主要追求目标。第二次世界大战以来，以原子能技术、空间技术和电子计算机技术的发展为主要标志的第三次技术革命对市场环境的变化产生了深远的影响。自动化生产线的的广泛应用极大地提高了劳动生产率，产品供应日益充足，并开始出现商品供大于求的现象。到了 80 年代，市场环境已经发生了显著的变化、卖方市场已经向买方市场转变，顾客在供求关系中占了主导地位。面对日益丰富的市场产品，顾客对市场的要求也越来越高，他们不仅要求能以最低的价格获得最好的产品，还希望得到最好的服务。在这种情况下，企业过去所处的那个相对稳定的经营环境已经不复存在。旧的运行机制和组织体制受到了强烈的冲击，企业不仅要注重产品的成本和产量，还要注重产品质量、上市速度、产品创新和客户服务。单纯的技术革新已经无法从根本上提高企业的

竞争力，企业呼唤新的现代管理方法的出现。在这种竞争环境下，提出了企业经营过程重组（Business Process Reengineering－BPR）的概念和方法。

1990 年，曾任美国麻省理工学院教授的米歇尔·哈默在《哈佛商业评论》1990 年 7-8 期中发表了名为《重组：并非自动化，而是重构》的文章[47]。几乎是与此同时，《斯隆管理评论》在 1990 年的夏季刊上刊登了托马斯·H·达文波特等人撰写的《新工业工程：信息技术与经营过程再设计》[46]，由此开始了企业经营过程重组方法的研究与应用实践。

顾名思义，经营过程重组意味着对企业的经营过程进行重新思考与再设计，以追求企业性能的突破性提高。达文波特在他的文章中定义经营过程重组为“在组织内部或组织之间分析、设计工作流程或过程”。 哈默认为重组是对企业过程进行彻底的改造，使得评价企业的某些关键指标，如成本、质量、服务和速度能获得显著的改善。与传统的基于分工理论的管理思想相比，BPR 具有以下三个显著的特征：

1） **面向顾客，强调顾客需求**

过去，顾客只是被动的产品使用者，厂家生产什么产品，顾客就接受什么产品。但是，随着卖方市场向买方市场转变，工业化时代大规模生产的企业模式宣告结束。激烈的全球化市场竞争，使得顾客由被动的产品接受者转变成了积极的产品决定者。企业必须全面考虑并满足顾客的个性化需求，及时响应市场的变化，才能赢得市场竞争。因此，顾客的需求成为企业实施 BPR 的最根本的驱动力。

2） **面向企业经营过程**

在传统的劳动分工原则下，企业的组织结构是按功能进行划分的，各职能部门把一个过程分为若干小块。过细的分工使得不同部门之间出现大量的合作与协调，即使人们能够运用先进的信息技术最优地完成每一小块的工作，也很难保证经营过程在整体性能上达到最优。同时，部门间大量的合作与协调使得经营流程人为地变得更复杂，工作效率更低下。因此，BPR 理论就是要彻底打破劳动分工理论的约束，跨越职能部门的条条框框，以经营过程为核心重建与经营过程相匹配的企业运行机制和组织结构，实现企业对全过程的有效管理和控制，使企业真正直接面对用户。

3） **追求企业性能的突破性提高**

BPR 追求的不是企业性能的渐进式提高或局部改善，而是期望通过对流程的重新设计，以提高顾客的满意度为主要方向，追求企业性能的飞跃，因此，有人把 BPR 称为“现代企业管理的一场革命”。

企业经营过程重组的概念问世以来，受到了企业界和理论界的广泛关注，欧美的多家企

业率先对现有的经营过程进行了改造。据资料显示，截至 1994 年底，有 75%－80%的美国大型企业对其经营过程进行了改造，仅 1994 年，美国各大型公司花费在经营过程重组上的人工费和咨询费用就超过 70 亿美元，若包括所投入的技术费用，则耗资高达 300 多亿美元。实施企业经营过程重组的企业中有不少已经取到了显著的成效，如福特汽车公司通过重组物料采购过程，使财务会计部的人从 500 多人下降到 125 人，而工作效率大大提高；数字设备公司（DEC）将 55 个财务部合并成 5 个，裁剪了 450 个工作岗位；CIGNARE 公司每年的运作成本减少了 150 万美元；美国电话电报公司（AT&T）下属的全球商务通讯系统部通过改造制造、服务和订单落实等流程，成功地扭亏为盈。

但是，并不是每一个进行经营过程重组的企业都能够收到令人满意的效果或实现预期的目标，有报告表明，约有 70%的经营过程重组项目归于失败，投资于过程重组的 320 亿美元中约有 200 亿没有发挥应有的作用[48]。只有那些在一定的条件下实施经营过程重组的企业才获得了成功。在 BPR 实施成功率不高的情况下，不少专家不再主张对企业的经营过程进行革命性的、激进的改造。虽然革命性的过程改造有可能大幅度的缩短业务过程的周期时间或降低成本，但其代价往往是非常高昂的，而且有时还会引发其他社会问题，导致组织的不稳定。因此，专家们强调要以更冷静的头脑对待 BPR。有些情况下对过程实施更稳妥的持续改进，也同样会取得良好的经济效益。

10.1.2 企业经营过程重组的实施步骤

企业实施 BPR 是一项非常复杂的工程，它往往会引起企业的文化、管理理念、组织结构和业务过程等多方面多层次的变化，因此，必须有步骤、有目的和有计划的组织实施，才能够保证经营过程改造的成功。实施企业经营过程重组的生命周期可以分为项目规划、过程建模、过程优化、系统实施和项目评价五个阶段， 如图 10.1 所示。

项目规划阶段的主要任务是制定企业经营规划（包括制定企业的近期和远期发展战略）和企业经营过程重组的目标。从以前实施企业经营过程重组的成功实例中，我们发现，企业的高层领导人通过制定企业的发展战略来从较高层次上指导项目的实施是非常有必要的。除此之外，企业领导人还应该对实施 BPR 的必要性和重要性形成一致的认识，确定过程重组的具体目标。一般来说，企业希望通过改造经营过程来降低生产成本、缩短生产时间、提高产品质量和服务水平、或提高客户的满意程度。

过程建模阶段的主要任务是准确的描述企业现在的业务过程，建立经营过程模型。过程

建模是经营过程分析和优化设计的基础，目前已有多种过程建模方法和工具可以用来帮助企业描述和分析经营过程。

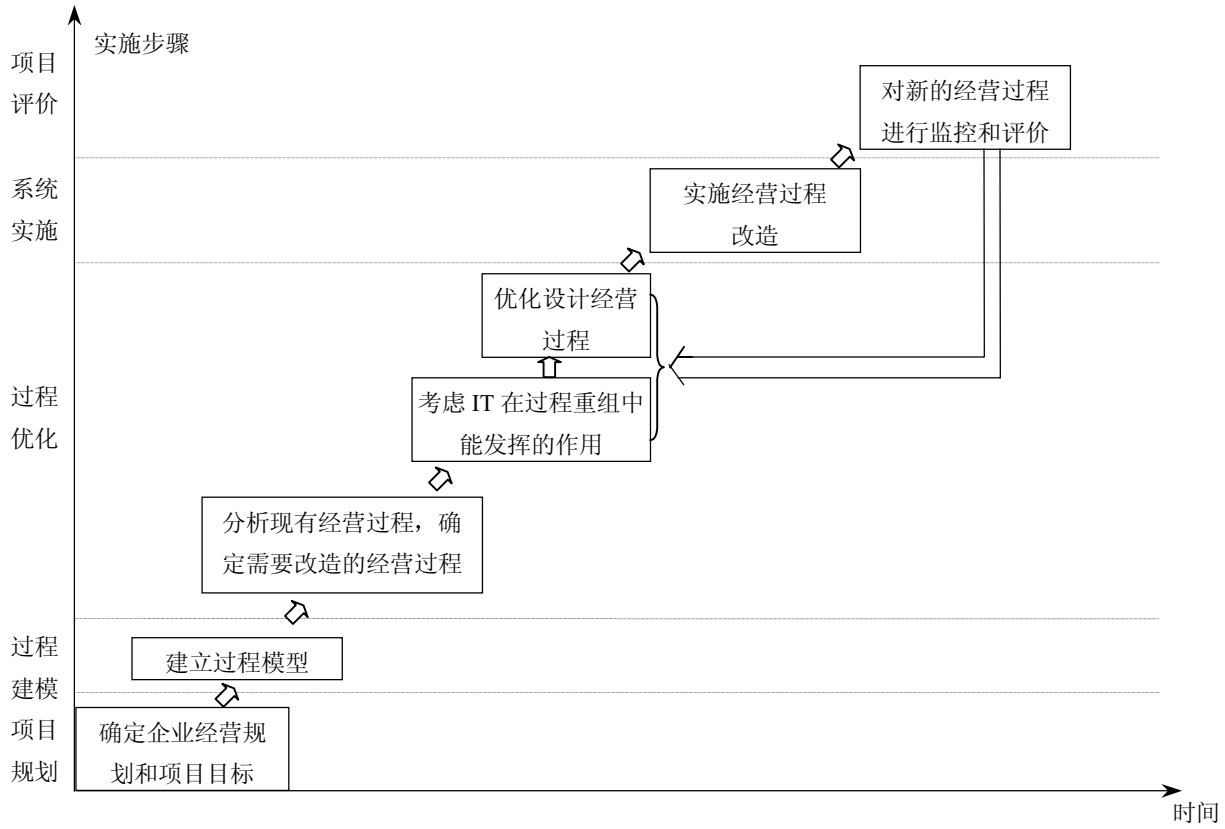


图 10.1 企业经营过程重组实施的五个阶段

过程优化是企业经营过程重组实施中一个非常重要的阶段，它的主要任务是在已建立的经营过程模型的基础上，分析和优化企业的经营过程。优化经营过程的时候，首先要考虑信息技术所能发挥的重要作用。以往，信息技术在工业工程中通常被认为是替代人力的自动化或机械化动力，其主要目的是提高企业事务处理的能力和速度。而现在，随着计算机技术、网络技术和通讯技术的迅速发展，信息技术已经渗透到企业的各个方面，包括办公自动化及客户服务领域。为了能够更好更快的响应顾客需求和输出产品，企业越来越需要具有更灵活的、面向工作组的和基于通信的协同工作能力。信息技术是 21 世纪的企业用来降低实现组织间协调而必须付出的代价的最有效的工具，而且，它能从根本上改变企业的行为方式。例如，当信息技术能够实现通过计算机网络远程交换计算机辅助设计图纸时，产品开发过程就会发生巨大变化：开发人员不再受到地域的限制，开发小组可以在计算机协同工作环境的支持，在异地协同设计一个产品，产品开发的并行水平大大提高，从而提高产品设计的效率和质量。由于信息技术能够在企业经营过程重组中发挥重要作用，因此，在实施过程改造之前，要充分考虑信息技术的所能发挥的能力，选择合适的技术运用到新的经营过程中去。

系统实施阶段的主要任务就是根据优化的过程模型，在企业中建立相应的信息系统，改

造原有的业务流程，并重新建立与新的业务过程相匹配的企业组织结构，进行人员培训，从而完成经营过程的改造。

当系统实施完成后，还要对新的经营过程进行监控和评价，以便发现其中存在的问题，在必要的情况下，还要对过程进行再次完善。

10.2 工作流管理技术在企业经营过程重组中的应用

工作流管理是近年来迅速发展起来的一项新兴技术。虽然工作流的概念最早起源于生产组织和办公自动化领域，但是随着对工作流管理技术研究的展开和深入、以及计算机网络技术和分布式数据库技术等辅助信息技术的迅速发展和成熟，人们越来越意识到，工作流管理是一种能够有效的控制和协调复杂活动的执行、以及人与应用软件之间交互的信息技术手段。因此，工作流管理技术的应用范围不断拓展，目前，已经成功的运用到图书馆、医院、保险公司、银行等多种行业。

虽然工作流管理系统可以有不同的应用范围和多种实施方式，但是从它的应用过程来考察，大多数工作流管理的应用过程都经历了过程定义（建立工作流模型）、过程优化、应用系统开发和工作流管理系统执行四个阶段，如图 10.2 所示。

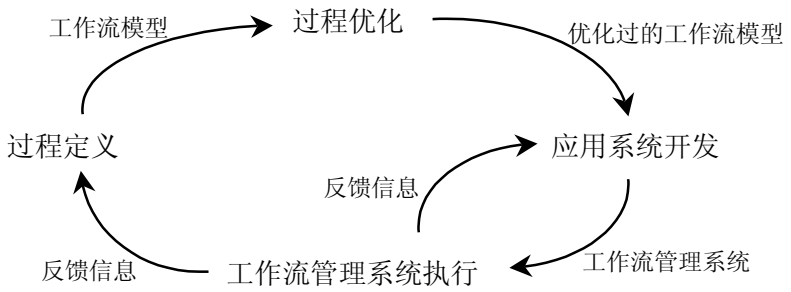


图 10.2 工作流管理系统应用的生命周期

1) **过程定义：**过程定义阶段的主要任务根据企业中的业务过程建立过程模型（工作流模型），也就是用计算机可处理的形式化定义来描述经营过程，它是分析过程与实现工作流管理的必要基础，过程建模工作主要由过程工程师来完成。
2) **过程优化：**过程优化是运用静态的或动态的模型分析方法和手段来分析已经建立的工作流模型的性能，发现模型中可能存在的瓶颈、死锁等问题，在此基础上改进和优化工作流模型。这一任务主要由过程分析工程师来负责完成。
3) **应用系统开发实施：**应用系统开发实施就是根据优化的工作流模型，建立相应的信息系统、数据库系统，实施工作流管理软件、实现工作流管理软件和其他应用系统的集成。

这个工作主要由 IT 工程师负责完成。

4) **工作流管理系统执行：**工作流管理系统执行是指采用工作流管理软件，根据优化的工作流模型创建过程实例、调度和控制业务过程中活动的执行。执行过程中，管理系统将完成计算机系统与操作人员的交互、以及触发相关的应用系统提供的功能。在工作流管理系统的试运行阶段，过程工程师和 IT 工程师要监控企业业务过程和工作流管理系统的运行情况，并根据运行监控结果对工作流实例的执行进行人工干预。

对比工作流管理系统应用的生命周期与实施企业经营过程重组的五个阶段，不难发现，工作流管理技术能够较好的支持过程重组中的过程分析、过程优化、系统实施和项目评价四个阶段。同时，由于工作流管理系统能够调度和监控过活动的执行，良好地实现过程管理与控制，因此，工作流管理技术经常是实施过程重组最广泛采用的技术。以工作流管理技术为基础，国外多家软件厂商开发了支持企业经营过程重组的软件产品。

本章的以下部分将详细工作流管理技术在企业中的应用方法，包括工作流模型建立、工作流模型的分析和优化、工作流管理系统的实施、以及任何采用工作流管理系统来管理和控制经营过程，最终实现企业经营过程的优化重组。

10.3 工作流模型建立

从第二章的概念介绍中，我们知道，过程建模是经营过程分析与经营过程重组的重要基础。它用计算机可以理解和处理的形式化定义准确地描述企业的经营过程，供流程分析和优化使用。在系统实施阶段，IT 工程师根据过程模型设计相应的信息系统，定义系统的功能构件配置，使得所建立的系统能够按过程实现横向集成，而不是按传统的部门或功能划分结构，从而满足企业核心价值流的要求。此外，过程模型还是记录和保存经营过程知识的一种有效途径，不同的组织或信息系统可以根据不同的需求访问过程模型，实现经营过程知识的共享。

工作流模型是过程模型的一种表现形式，目前已有多家研究机构和工作流管理软件生产商对工作流模型的描述语言和定义方法进行了研究，产生了多种不同形式的工作流模型，例如，基于事务模型的工作流模型、基于 Petri 网的工作流模型、基于语言行为的工作流模型、和基于活动网络的工作流模型等。不同形式的工作流模型其建立方法也有差异。概括的说，建立工作流模型的方法学主要可以分为以下三类：

1） **基于通讯（Communication-based）的工作流建模方法：**该方法以描述顾客与执行者之间的对话和信息交互为基础，定义工作流中的每一个动作，从而建立工作流模型。这种过

程建模方法被 Action 技术公司所采用并开发了相应的工作流产品—Metro（详见第五章）。

2） **基于加工对象（Artifact-based）的工作流建模方法：**该建模方法的描述对象是在过程中创建、修改和使用的加工对象。也就是说，通过描述工作产品以及它们在业务过程中的路径来建立工作流模型。该方法适用于为制造过程建模。

3） **基于活动（Activity-based）的工作流建模方法：**基于活动的过程建模方法是应用最为广泛的一种方法，它通过把经营过程分解为一系列存在一定逻辑关系的活动（任务），然后描述每一个活动的属性以及活动间的相互关系来定义工作流模型。基于 Petri 网的工作流模型、基于活动网络的工作流模型和 CIMFlow 模型等都是采用这种方法建立的。本节也将采用基于活动的工作流建模方法来为企业建立工作流模型。

10.3.1 企业工作流模型的组成

企业是一个非常复杂的社会、经济、物理的系统，必须从多个层次多个角度来考察才能全面地描述经营过程。通常，可以从组织、资源、业务流程和信息这四个方面来考察企业的经营过程，建立相应的组织视图、资源视图、过程视图和信息视图。

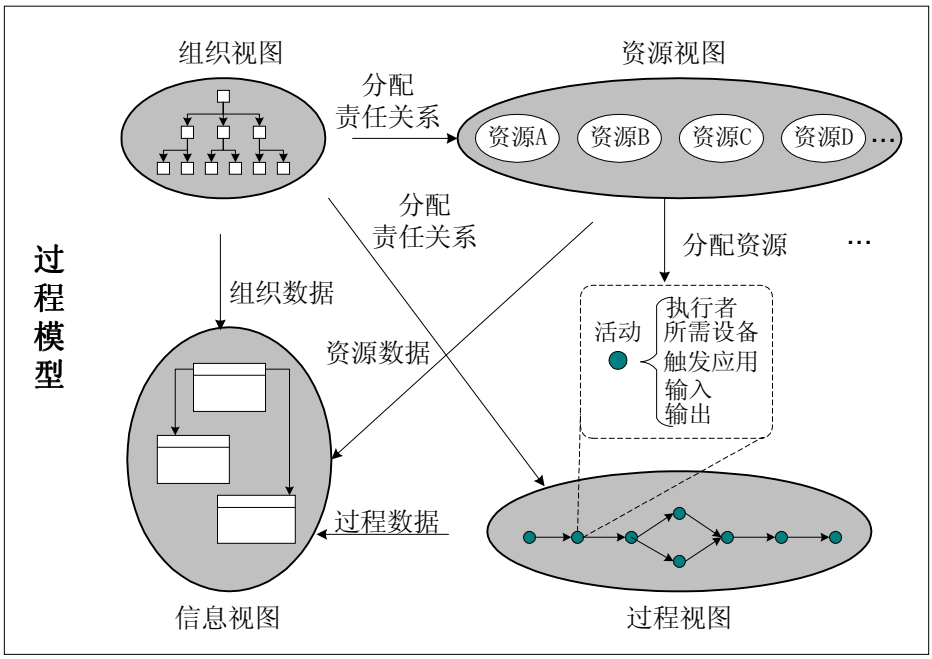


图 10.3 工作流模型的组成

如图 10.3 所示，一个企业的工作流模型由组织视图、资源视图、过程视图和信息视图组成。每一个视图都从不同的侧面描述经营过程的信息，而且这些视图之间存在着非常密切的关系。

1) **组织视图**

组织视图描述企业的组织单元和组织单元间的关系。组织单元是具有一定功能和责任的组织实体。组织单元之间往往存在着从属或协作关系，形成一定的层次结构。从较高层次的意义上说，整个企业就是一个组织单元，它由一定数量的雇组和其他资源组成，能够完成一定的生产任务或者对外提供一定的服务，同时承担一定的社会责任。从较低的层次来看，企业中每一个部门也是组织单元，它们是企业的下级组织单元，完成业务过程中的某一部分，承担一部分厂内的或社会的责任。

2) **资源视图**

资源视图描述企业中资源的类型以及资源实体的属性。资源是工作流模型中非常重要的一个概念，它是活动可以执行的必备条件。资源实体可以是活动的执行者（人力资源）、执行活动所需的设备、物料、或者是活动执行后产生的新的物理实体。组织视图和资源视图之间存在着映射关系，即每一个资源实体都有与其相对应的责任组织单元，由该组织单元负责对此资源实体的使用和维护。

3) **过程视图**

过程视图描述企业的业务流程，它定义业务过程中包含的活动以及这些活动之间的逻辑关系。活动和活动间的连接弧是过程视图中最基本的组成元素。通过描述活动的基本属性，如活动由谁来执行、有哪些部门参与和负责、活动执行需要什么设备或原材料等，可以建立过程视图与组织视图和资源视图的关联。

4) **信息视图**

信息视图从信息关系的角度描述了经营过程中的数据结构特征和数据关系。信息视图的信息来源于组织视图、资源视图和过程视图中的数据结构及数据关系，它的建立是数据库设计的基础，也为建立管理信息系统、实现计算机辅助企业管理提供了技术保证。

10.3.2 建立工作流模型的步骤

工作流模型包括过程、资源、组织和信息四个视图，为了能够完整地描述企业的经营过程，同时还能保证视图之间的一致性，建立工作流模型时必须遵循一定的步骤。虽然不同企业的业务过程类型和特点不同，不同的建模软件对建立工作流模型的步骤也有不同的要求，但是，由于工作流模型的不同视图之间存在着明确的依赖关系，建立工作流模型的步骤还是有规律可寻的。下面，本节采用基于活动的工作流建模方法，介绍如何为企业建立工作流模

型，其建模过程可以分为收集过程数据、建立企业经营过程的较抽象的过程视图、建立企业的组织视图、建立企业的资源视图、定义活动的细节信息、和建立企业的信息视图这六个步骤。

步骤 1：收集过程数据

建立工作流模型的第一个步骤是由过程工程师收集经营过程数据。通常，过程工程师与企业的各级领导者和任务的实际操作者进行会谈。过程工程师提出问题，企业领导者或操作人员回答问题，过程工程师从问题的答案中获取过程信息。由于建立的工作流模型是否能准确的描述企业的经营过程取决于收集到的数据是否全面而且正确，因此，过程工程师要精心设计会谈中要提出的问题，使问题的答案尽可能的包含企业经营过程的各方面的信息。收集过程数据时提出的问题可以分为两大类。

第一类问题是关于企业组织和资源信息的问题，包括：

1) 什么是影响企业成功的关键因素？
2) 企业现在面临的挑战是什么？未来它还将面临怎样的挑战？
3) 企业的组织结构是怎样的？
4) 根据企业的性质和特点，评价企业经营过程的主要性能指标是什么，例如，是过程时间、产量、成本、还是产品的质量？
5) 哪些人参与了经营过程中活动的执行？付给他们的工资（成本）是多少？
6) 每一个组织单元负责管理哪些资源？这些资源在经营过程中扮演的角色是什么？他们的工作会对其他的资源产生什么影响？
7) 资源个体之间存在着什么关系？

第二类问题是关于业务过程运作的问题，包括：

1) 企业主要的业务过程是什么？例如，是完成客户订单、产品开发、还是处理顾客的请求？
2) 根据影响企业成功的关键因素，哪些过程是具有竞争力的，而哪些过程需要进一步的改进？
3) 企业的政策如何对经营过程的运行产生影响？
4) 哪些事件可以驱动经营过程的运行？在给定的时间段内，经营过程的产出是多少？
5) 每一个活动的输入和输出分别是什么？
6) 对于每一个主要的业务过程来说，它所期望实现的结果或产出是什么？
7) 活动之间的逻辑顺序是怎样的？

8） 业务过程中，哪些活动是已经实现了自动化的活动？

9） 资源参与了哪些活动的执行？在一定时间段内，活动的产出是多少？

10）每一个活动的可以执行的条件是什么？执行时间是多长？

11）对决策活动来说，每一个决策分支发生的概率是多少？

过程工程师可以根据实际需要设计数据表格，将收集到的信息进行归纳整理，填入表格。这可以帮助过程工程师更好的理解经营过程中的每一个活动。例如，数据表格可以是如表 10.1 形式的表格。

表 10.1 数据收集工作表

过程/活动	资源描述	持续时间	输入物料	输出物料	条件

除了举行会谈外，过程工程师还可以从其他途径获得过程信息，例如过程工程师可以从以下文件中获得许多有价值的信息：

1） 企业的战略商业计划。

2） 企业的年报。

3） 企业的过程手册或工艺手册，或者是以前建立的过程模型。

4） 企业中现在正在使用的各种表格。

5） 企业中的所有报表。

6） 企业中已经建立的信息系统的信息。

步骤 2：建立企业经营过程的较为抽象的过程视图

在完成企业过程数据的收集之后，过程工程师就可以对信息进行分类整理，并着手建立工作流模型。首先，过程工程师要根据实际需要选择适合的建模工具。目前，有多种软件产品都能支持过程建模和过程分析，如 IBM 的 Business Process Modeler、Action 技术公司的 Process Builder、Holosofx 公司的 Workflow BPR 、Interfacing 公司的 FirstSTEP 、MicroGrafx 公司的 iGrafx、和 CACI 产品公司的 SimProcess 等。其中，有许多过程建模工具已经提供了与相应的工作流管理软件的集成接口，如 IBM 的 Business Process Modeler 能与 IBM 的工作流管理系统 FlowMark 集成、Workflow BPR 定义的过程模型能够直接输出到 FlowMark 和 FileNet 公司的工作流产品 Visual Workflow 中。不同过程建模和分析软件选用的建模语言可能不同，其应用范围和特点也不同，过程工程师要根据企业的性质和过程建模的目的选择合适的建模工具。

选择了所要使用的工作流建模工具之后，过程工程师就可以把企业的主要业务流程分解成若干活动或子过程，建立较抽象的经营过程的过程视图。一个企业往往包含多个业务过程，如一个生产型企业通常包括产品设计过程、生产计划的生成过程、原材料采购供应过程、库存管理过程、生产过程、产品销售过程、财务结算过程等。每一个业务过程都由若干逻辑相关的活动组成。根据建模抽象层次的不同，一个业务流程中又可能包含一个或多个子过程。子过程实际上就是一个局部的过程视图，它也由若干有逻辑联系的活动构成，以实现某种功能。子过程的引入可以增强过程视图的表达能力，使过程视图具有了层次化的结构。建立抽象的过程视图时要解决的主要问题是：

1) 划分企业中的业务过程，明确企业中有哪些主要的业务流程，它们之间的关系如何；

2) 明确每一个业务流程实现的功能或目的；

3) 确定每一个业务流程的输入和输出，包括信息的、和物料的输入和输出；

4) 确定每一条业务流程中包含的活动或子过程的名称、及完成的功能；

5) 确定每一个活动的输入和输出，包括信息的输入和输出，物料的输入和输出；

6) 确定活动之间的逻辑关系。

抽象的过程视图描述了企业的过程目标和主要业务流程。它可以帮助人们理解企业中不同的业务过程是如何相互影响相互作用的，每个业务过程又有哪些活动或子过程构成。

例如，假设有一家生产型企业 A，它的生产特点是以销定产，根据销售合同来组织生产。它的业务过程简单描述如下：企业承接生产订单后，首先根据产品设计信息以及企业的生产能力制定生产计划。然后将生产计划同时下达给原材料采购部门和生产部门。原材料采购部门根据生产计划、产品设计部门提供的产品 BOM 和原材料的库存信息提前采购生产所需的物料，以保证生产的正常进行。采购得到的物料送入仓库进行统一管理。生产车间根据生产计划和产品设计图纸来组织生产，各车间直接到仓库领取生产所需到原材料。生产出的产品由销售部门负责销售，并向用户提供售后服务。整个原材料采购、原材料管理、生产、销售及售后服务中的费用均由财务部门进行结算。

库存管理过程还包括以下环节：① 采购回来原材料首先送到质检部门进行质量检验。如果质量合格，则办理入库手续；如果不合格，则办理原材料退货手续；② 当车间需要用料时，车间向仓库提出申请，仓库根据车间的用料申请办理原材料出库单，原材料出库。根据原材料的入库单和出库单可以统计现有的原材料库存信息；③ 入库单和出库单的复件将交给财务部门进行财务结算。

对于 A 企业的业务过程，为了使描述更加清晰，过程工程师把其 划分为产品设计过程、

产品生产过程、原材料管理过程和财务结算过程四个主要的业务过程，其中，产品生产过程由“接受订单”、“制定生产计划”、“生产”和“销售和售后服务”四个子过程组成，原材料管理过程由“原材料采购”和“库存管理”两个子过程构成，如图 10.4 所示。这四个经营过程之间存在着密切的物料或信息交互，例如，产品设计过程要为生产提供详细的产品图纸，为物料采购提供产品 BOM；原材料管理过程要向生产子过程提供全部的所需原材料，而产品生产过程和原材料管理过程中的所有发票、支票和入/出库信息都要反馈给财务结算过程进行财务结算和成本核算。

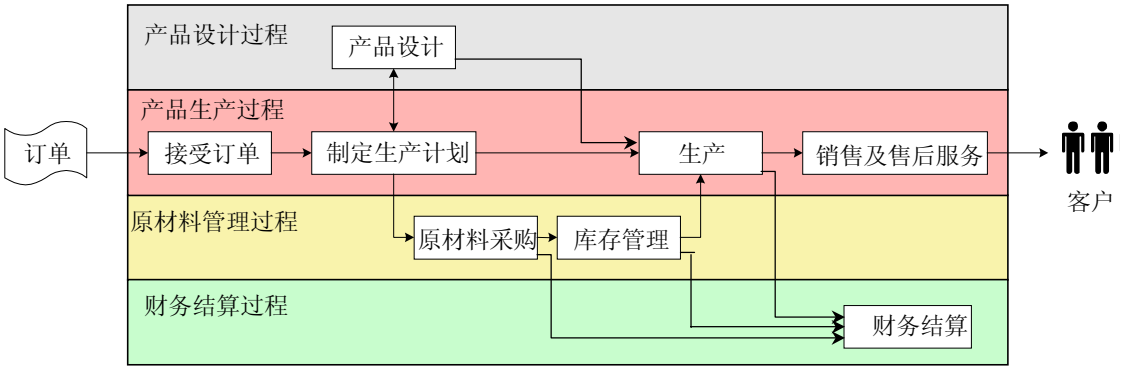


图 10.4 企业 A 的抽象过程视图

根据对库存管理过程的描述，过程工程师把库存管理子过程分解为以下活动，如图 10.5 所示。

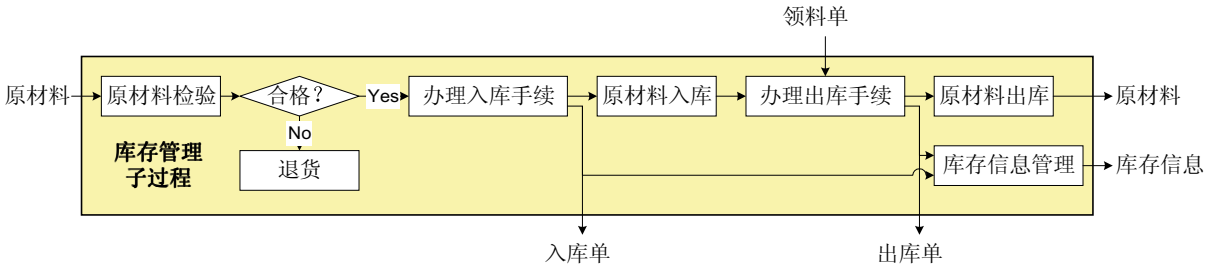


图 10.5 库存管理子过程实例

步骤 3：建立企业的组织视图

组织视图由面向企业职能部门的静态组织结构组成。过程工程师根据会谈的结果和企业的组织机构图，定义组织单元的属性和组织单元之间的关系，建立组织视图。建立组织视图时过程工程师需要定义的内容有：

1） 组织单元的名称、主要功能、地理位置和描述；

2） 上级组织单元名称；

3） 下级组织单元名称。

图 10.6 给出了一家企业的组织视图实例。为了在说明问题的前提下使图的结构尽可能的简单清晰，图中只给出了生产管理科的说明。对其他组织单元的说明类似。

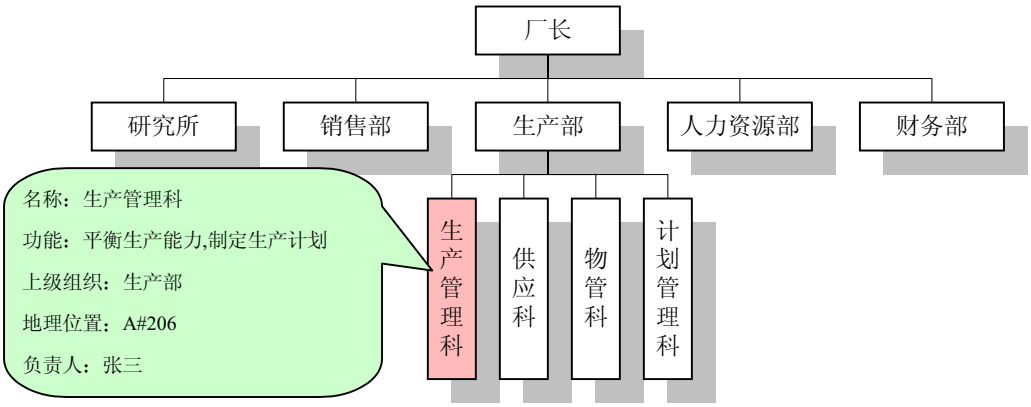


图 10.6 企业 A 的组织视图

步骤 4：建立企业的资源视图

企业的资源视图定义了资源类型、资源实体和资源池的属性。资源是活动可以执行的必要条件，活动执行过程中可以调用资源视图中定义的资源。资源类型是从资源分类的角度描述企业资源（如人和设备分属于两类不同的资源）。资源类型可以嵌套定义，子资源类型对象可以继承父资源类型对象的属性，从而构成企业的资源分类树。资源实体是指原子级的具体资源。而资源池则是从地理位置的角度或应用的角度出发来描述资源，一个资源池是处于同一地理位置上的或为同一项应用服务的若干个同类资源实体的集合。

建立资源视图时，过程工程师需要定义：

1) **资源类型的属性：**包括定义资源类型的名称、功能、描述、负责该资源类型的组织单元的名称、和父资源类型的名称。
2) **资源实体的属性：**包括资源实体的名称、所属的类型、功能、特性描述、负责该资源的组织单元的名称。
3) **资源池的属性：**包括资源池的名称、描述、地理位置、负责该资源的组织单元的名称、所包含的资源类型的名称和资源实体的名称、资源池容量和资源池的域值。
4) **资源的单位成本。**

当定义了资源类型、资源实体或资源池的组织属性“负责该资源的组织单元的名称”之后，就建立起了资源视图到组织视图的映射关系。这种映射是多对多的映射，即一个组织单元可以对多种资源型、多个资源池或资源实体负责，而一种资源类型、资源池或资源实体也可以分属于多个组织单元 。

图 10.7 给出了一个简化的资源视图的实例。

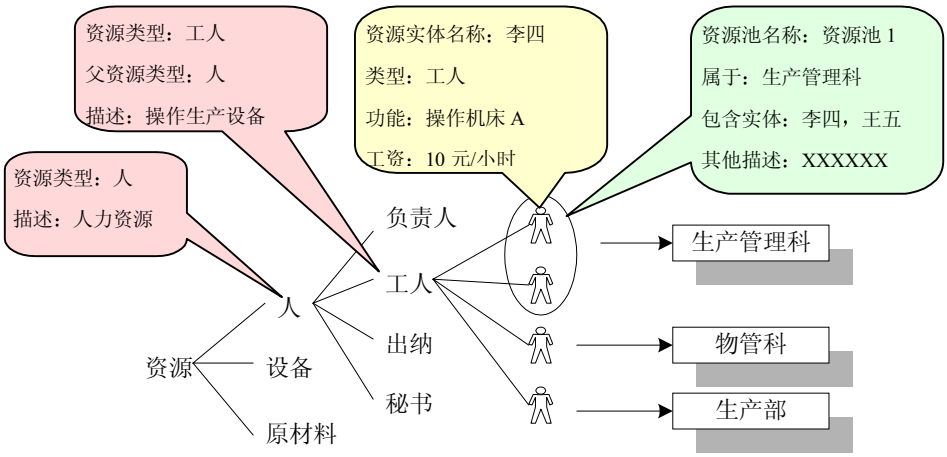


图 10.7 资源视图实例

步骤 5：确定对过程/活动负责的组织单元

在步骤 2 中，过程工程师已经建立了比较抽象的过程视图，视图中定义了过程/子过程、以及过程/子过程中活动的名称、功能及活动间的关系，但并没有定义过程或活动是由哪些部门来负责完成的、活动的执行需要哪些资源。因此，在本阶段，过程工程师就要开始定义活动执行所需要的信息，指定对过程或活动的执行负责的组织单元，为活动的执行分配资源，确定活动可以开始执行或停止执行的条件，描述活动的具体行为。

过程工程师需要定义的过程和活动细节包括：

1） 对过程、子过程或活动负责的组织单元的名称；

2） 活动执行所需的资源名称以及需求数量；

3） 活动的输入和输出；

4） 活动执行的开始条件和结束条件；

5） 活动的执行时间；

6） 活动执行的其他经营约束；

7） 若是决策型活动，定义决策条件或决策决策结果的发生概率。

例如，对活动“把发票信息录入到信息系统中”的描述如下图 10.8 所示：

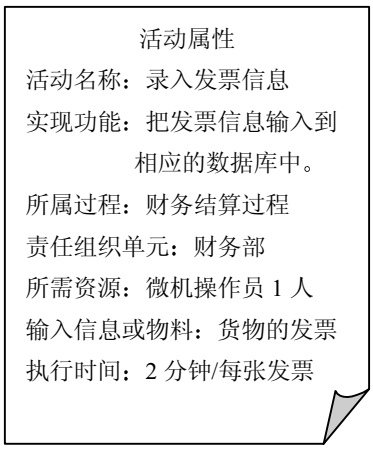


图 10.8 活动描述

类似地可以描述经营过程中每一个活动的属性，从而建立完整的过程视图。

步骤 6：建立信息视图

信息视图描述了企业经营过程中的数据结构和数据关系，构成信息视图的数据来源于组织视图、资源视图和过程视图中定义的相关数据、文档、属性等。信息视图的建立是数据库设计和实施管理信息系统的基础。由于过程工程师了解经营过程中的信息流，而系统工程师对如何设计数据库和信息系统有丰富的经验，因此，建立经营过程的信息视图需要过程工程师和系统工程师进行协同工作。

值得一提的是，信息视图并不是每一个工作流模型所必须包含的部分。针对不同的应用需求，在工作流模型中也可以不单独列出信息视图这个部分，而将涉及的相关数据和信息定义在过程、组织、或者资源模型中就可以满足需要了，如建立过程模型的目的是为了分析评价经营过程的性能，而不是为了最终实施工作流管理系统，那么，就不需要建立信息视图。在这种情况下，过程视图、组织视图和资源视图就已经构成了完整的工作流模型，可以供过程分析和评价使用。

10.3.3 建立企业工作流模型的实例

本节对一家软件公司，采用 Micrografx 公司的 iGrafx 软件建立其业务流程的工作流模型，并通过这个例子来说明建立企业工作流模型的步骤和方法。

例 1：软件公司 A 根据订单为顾客提供所需的软件产品。该公司共有五个部门，它们分别是：销售部、信用认证和财务部、生产管理部、软件复制部和包装运输部。其中，销售部

有 3 名雇员，信用认证和财务部有 5 名雇员，生产管理部有 2 名雇员，软件复制部有 2 名雇员，包装运输部有 5 名雇员。

该公司的主要业务流程描述如下。为了描述的方便，我们将在方括号[ ]内标明雇员执行活动所需的时间。

1） 首先由销售部的一名雇员接受顾客提出的订单[2 个小时]。

2） 接受顾客订单后，销售人员把订单转交给信用认证和财务票部。当一名雇员接收到订单[5 分钟]后，首先要审查顾客的信用状况是否良好[2 天]。有 90％的可能是顾客的信用良好，那么就可以准备发票[2 小时]，同时，订单将被转给生产管理部。当确认产品托运之后，就可以寄出发票[1 天]。而另外 10％的可能是顾客的信用状况不够好，那么则交给 1 个销售人员进行协调处理[1 天]。经过销售人员的协调，有 50％的可能是顾客的信用问题仍然无法解决，则取消该订单。而另外 50％的可能是顾客的信用问题得以解决，则销售人员通知认证人员顾客的信用良好、可以为该顾客开发票。

3） 生产管理部的一名雇员接到订单[1 小时]后，首先判断库存中是否有该项产品。库存中有该产品或没有该产品的可能性各是 50％。

4） 如果该产品有库存，那么生产管理部直接通知包装运输部，由包装运输部的一名雇员核对订单[30 分钟]，然后把软件产品邮寄给顾客[1 天]。

5） 如果该产品没有库存，则由软件复制部的一名雇员进行生产调度[15 分钟]，并复制磁盘（生产软件）[2 小时]，再交给包装运输部，由包装运输部的一名雇员包装软件[3 小时]，然后再邮寄给顾客。

在了解了企业的背景和主要的业务流程之后，过程工程师选用 iGrafx process 软件来建立企业的工作流模型。iGrafx process 是 MicroGrafx 公司开发的过程建模和仿真软件，它能够为用户提供友好的用户界面、多样的模型元素、强大的仿真能力和灵活的仿真报表生成样式[50]，使过程工程师可以容易地建立企业的工作流模型，并对其进行仿真分析。

根据本书中提到的建模步骤，结合 iGrafx process 软件的具体要求，过程工程师首先确定企业 A 的主要业务过程所包含的活动，同时确定这些活动由哪些组织单元负责执行，建立工作流模型的过程视图，如图 10.9 所示。然后，过程工程师描述企业的资源，建立工作流模型的资源视图，如图 10.10 所示。这些资源稍后将被分配给过程中的每一个活动。

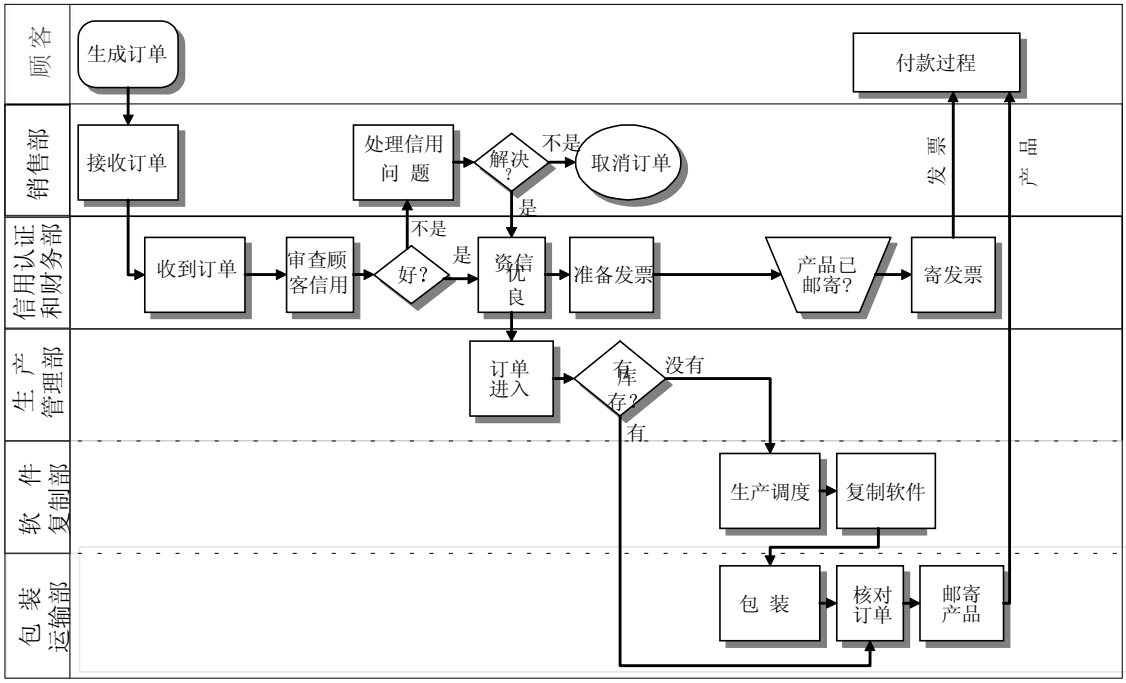


图 10.9 企业 A 的过程视图

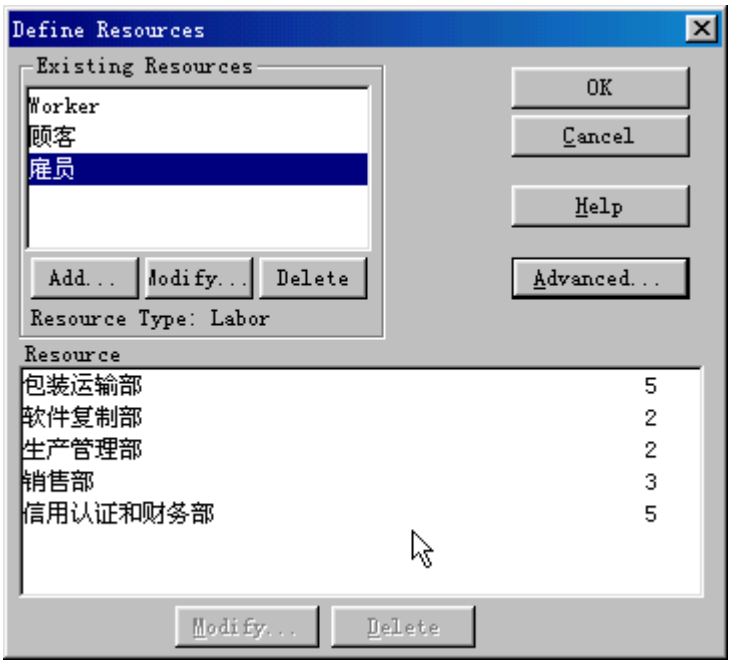


图 10.10 企业 A 的资源视图

然后，过程工程师在活动的属性窗口中定义每一个活动的输入和输出、活动执行所需的资源、活动执行的时间、活动的一般信息、及其他由用户定义的活动信息，图 10.11 给出了活动“接收订单”的属性窗口。至此，就建立了完整的过程模型，可以供用户分析使用。

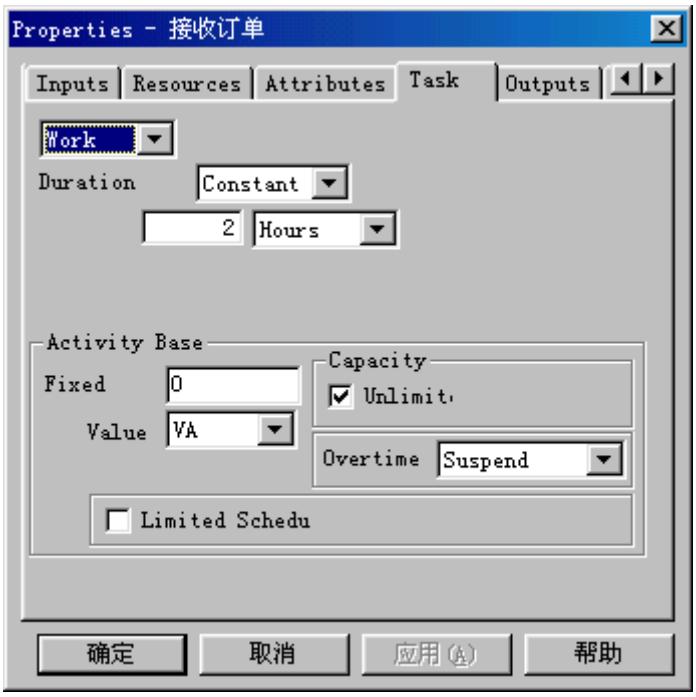


图 10.11 活动的属性窗口

10.4 企业工作流模型的分析和优化

企业经营过程重组不是简单的用各种先进的自动化技术和信息技术来实现经营过程的自动化，而是要通过改造原有过程中存在的不合理因素来提高经营过程的关键性能指标，因此分析经营过程、发现其中存在的问题并对其进行优化是企业实施 BPR 中非常重要的一个环节。由于工作流模型比较全面的反映了经营过程的结构和动态行为，因此通过分析工作流模型的性能来对经营过程进行诊断是一种可行的、有效的、和经济的手段。

目前，有多种方法可以用来分析工作流模型，比较常见的有“what-if”分析法和仿真分析法。

“What-if”分析法，顾名思义，就是先假设某些条件成立或者假设某个经营事件发生，在这种情况下分析活动的执行情况以及工作流模型的运行性能。“What-if”分析法可以有效的分析简单过程，发现模型中存在的瓶颈、死锁等问题，但是当一个企业系统非常复杂，多个经营过程互相关联时，“What-if”分析法就显得能力不足。

企业的业务过程是一个复杂的、存在很多不确定因素的系统，因此，几乎不可能找到一种解析的算法对它们进行描述和分析，这种情况下，工作流模型的仿真分析方法就是一种可行的和有效的分析手段。工作流模型仿真的目的是定量地分析企业经营过程运行的各项性能指标，如过程的运行时间、运行成本、和资源利用率等，判断经营过程中是否存在瓶颈或死

锁因素，经营过程的运行性能是否良好。仿真结果可以为经营过程的评价、过程改进方案的可行性和有效性验证的依据，也是企业进行决策的可靠的基础。

由于企业经营过程具有离散事件系统的典型特征，这类系统中的状态只是在离散时间点上发生变化，而且这些离散时间点一般是不确定的。例如，一个银行，假设上午 9 点 15 分开门，下午 5 点 15 分关门，在这一段时间内，顾客的到达时间是随机的，为每个顾客的服务时间长度也是随机的。由于工作流模型描述的是企业经营过程，因此对它的仿真属于离散事件系统仿真。在离散事件系统中，某一次仿真运行得到的结果只不过是随机过程的一次取样，因而需要多次运行仿真，并对多次仿真的结果进行统计，所得到的统计意义下的仿真结果才具有参考价值。下面，将首先介绍离散事件系统仿真的一些基本概念，然后介绍如何对工作流模型进行仿真，并分析和优化工作流模型。

10.4.1 离散事件系统仿真的基本概念

离散事件仿真系统中，实体、事件和活动是三个最基本的概念[51]。

1) **实体：**实体是仿真系统中定义的物理对象。根据实体在仿真系统中的性质和作用，可将实体分为两大类。第一类是临时实体，它在系统中只存在一段时间，从系统外部到达系统，通过系统，最终离开系统，如制造过程中的物料。另一类是持久实体，它在相当长的时间内（相对于企业的一个经营过程而言）保留在系统中，是系统处于活动的必要条件，如制造过程中的各种设备。临时实体按一定的规律到达系统，在持久实体的作用下通过系统，最后离开系统，使整个系统呈现出动态性。

2) **事件：**事件是引起系统状态发生变化的事情。例如在银行系统中，顾客接受完服务就是一类事件。顾客在完成接受服务后，服务员的状态由“忙”转为“空闲”。从某种意义上说，离散事件系统就是由事件来驱动的。在一个系统中，往往存在多类事件，事件的发生一般与某一类实体相联系，而且还可能会引起别的事件发生，或者是另一类事件发生的条件。在仿真系统中，除了要定义系统中固有的事件外，还要定义所谓的“程序事件”，用于控制仿真进程。例如，对于银行系统上午 9:15 到下午 5:15 这一段时间内的动态过程仿真，就可以定义“仿真时间达到 8 小时后停止仿真”为一个程序事件。

3) **活动：**离散事件系统中的活动，可以理解为一段时间上的动作或行为。活动的执行往往标志着系统状态随着时间的变化而发生转移。通常，一项活动因为某个事件开始，而活动结束时又产生另一个事件。

仿真系统中还定义了进程的概念，一个进程由若干事件和活动组成。进程描述了它所包括的事件及活动间的逻辑关系及时序关系。例如，一份订单下达，经过原材料准备、多道工序生产、直到生产出最终产品并交到用户手中的过程，可称为一个进程。

除此之外，仿真系统还需要定义仿真钟。仿真钟用于表示仿真时间的变化。离散系统中，由于事件发生时间的随机性，仿真钟的推进步长是随机的，而且可以跨越没有事件发生的“不活动”周期，从一个事件发生时刻推进到下一事件发生时刻，推进呈跳跃性，推进速度具有随机性。

离散事件系统中，某一次仿真运行得到的结果只不过是随机过程的一次取样，因而需要多次运行仿真，并对多次仿真的结果进行统计，所得到的统计意义下的仿真结果才具有参考价值。

10.4.2 工作流模型仿真的应用范围

工作流模型仿真是利用离散事件驱动的仿真引擎模拟执行工作流模型中的各项活动，自动的推进工作流实例。经过多次仿真运行，得到一系列关于工作流模型运行的统计数据，如过程时间、过程成本、和资源利用率等。用户可以在这些仿真统计数据的基础上进一步分析和评价企业经营过程的各项性能。

仿真分析法具有以下优点：

1) **运行时间短、效率高：**一个需要运行几天或者几个月才能完成的经营过程，其仿真过程可能只需几个小时甚至几分钟。
2) **安全可靠，风险小：**工作流模型仿真是通过设置仿真环境、修改工作流模型和仿真初始化条件来模拟业务过程的变动和市场环境的变化，它不会对实际的业务过程的运行产生任何影响，避免了改变实际的业务过程而可能带来的不可挽回的影响。
3) **经济：**工作流模型仿真过程中，除了仿真系统的运行消耗外，不需要消耗其他的企业资源，因此，仿真运行成本低，非常经济。

通过工作流模型仿真能够计算和分析企业经营过程的多项性能指标，它可以：

1) 帮助企业制定资源能力计划，包括计算或验证合理的资源需求、验证增加或减少资源对经营过程的运行带来的影响。
2) 分析经营过程的运行时间。“时间”是评价经营过程运行性能的关键指标之一，分析经营过程的各种时间指标，如过程周期时间、等待时间、增值活动时间、和非增值时间等，

对于评价经营过程性能的好坏具有非常重要的意义。如业务流程的运转周期时间短意味着产品的上市时间短、表明企业能够快速的相应顾客的要求为顾客提供满意的服务；增值活动时间占运转周期时间的比例高说明业务流程的效率越高，流程比较优化。

3) 进行基于活动的成本分析，如判断经营过程中哪些活动的成本最高，哪些活动占用了大量资源而给经营过程带来的效益却很少，通过成本分析进一步寻求降低经营过程成本的途径；

4) 分析经营过程的吞吐量。吞吐量是衡量经营过程的业务负载能力的一项指标。吞吐量高意味着在一定的经济约束条件下，经营过程可以生产出更多的产品（产量更高）或者能够为更多的客户提供服务。

5) 分析经营过程中存在的瓶颈因素。执行时间长的活动或者利用率非常高的资源都容易形成过程中的瓶颈。通过分析过程仿真得到的统计数据，可以知道过程中哪个活动的执行时间最长，哪些资源的利用率偏高，从而分析过程中的瓶颈。

6) 为企业决策提供重要的依据。仿真数据从统计意义上反映了的经营过程的运行指标，它能够为企业决策提供可靠的依据。

7) 验证经营过程模型。设置与真实的经营过程运行环境相近的仿真环境及仿真初始化条件，在这种情况下比较仿真结果与经营过程的实际运行结果是否一致，可以验证该模型是否准确的反映了经营过程的主要特征。如果工作流模型不能准确的描述经营过程，则要对其进行修改。

8) 比较不同的过程改进方案。企业是一个复杂的系统，改变企业经营过程往往会对企业的文化、管理思想、组织结构和业务流程等多个方面产生重要影响，一旦过程改造失败，很有可能会对企业经营造成严重的负面影响，具有较大的风险性。而通过工作流模型的仿真分析，可以安全、经济、高效的分析和比较不同的过程改造方案可能带来的影响，验证方案的可行性、有效性、和合理性，大大降低的过程改造的风险，提高了改造的成功率。

10.4.3 工作流模型仿真的步骤

如图 10.12 所示，执行工作流模型仿真的过程可以分为设置仿真环境、执行仿真、结果分析三个阶段。

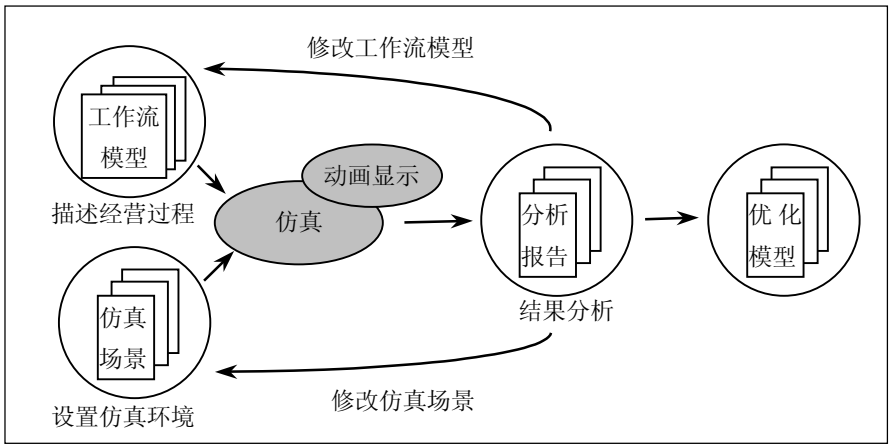


图 10.12 工作流模型仿真过程

1 设置仿真环境

设置仿真环境就是定义仿真场景。场景是一组与仿真运行有关的数据，场景的设置包括仿真运行设置、事务生成器设置、时间表定义、和其他的仿真选项。

1) **仿真运行设置**

仿真运行设置的主要任务是定义仿真的开始时间和结束时间，以及指定仿真报告的生成路径和文件名。从前面的介绍中，我们已经知道工作流模型的仿真属于离散事件系统的仿真，仿真系统中，除了要定义系统中固有的事件外，还要定义所谓的“程序事件”，用于控制仿真进程。定义仿真的开始时间和结束时间就是定义仿真的程序事件，例如，定义一次仿真运行的开始时间是 2000 年 5 月 1 日，定义结束时间是 2000 年 6 月 30 日，那么此次仿真就是模拟这一段时间内工作流模型的运行。当仿真时间到达 2000 年 6 月 30 日时，仿真运行终止。而指定仿真报告的生成路径和文件名则告诉系统该在什么路径下生成仿真报告，报告的文件名是什么，以方便用户的查找。

2) **事务生成器设置**

事务可以理解为来自于外界的工作流模型的驱动力，它一般作用于工作流模型的第一个活动，或者是作用于模型中的其他外界信息进入口。例如，对 10.3.3 例 1 的工作流模型来说，顾客提交了一份订单就是外界（顾客）产生了一个事务，驱动工作流模型中后序活动的执行。在实际的工作流管理系统中，事务是由外界（如顾客、其他的工作流管理系统等）产生的，而在仿真过程中，事务都由仿真系统产生，因此，定义仿真环境时要设置事务生成器，使仿真系统能够根据事务定义自动产生的事务，推动仿真的运行。事务生成器设置的不同会对仿真结果产生重要影响。一般来说，设置事务生成器时要确定事务产生的规律、数量。例如，假设顾客提交一份订单是某一个工作流模型的驱动事务，平均每 10 天顾客就会提交一份新

的订单要求定购一件产品，那么，仿真系统中可以定义如图 10.13 的事务生成器：

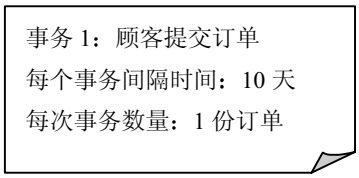


图 10.13 事务生成器的定义

3) **时间表定义**

在企业中，资源的使用遵循一定的工作时间表。例如，员工按照作息时间上下班，因此，员工只在上班时间执行活动，下班之后不执行活动或是加班执行活动。为了使仿真结果更加真实可信，设置仿真环境的时候也应该根据企业的实际情况为资源定义工作时间表，确定活动可以被执行的时间区段。例如，定义员工 A 的时间表是每周从周一到周五工作 5 天，每天的工作时间是从早上 9:00 到下午 5:00，节假日休息。那么，当仿真时间处于工作时间表之内（如周一上午 10 点），且员工 A 的状态为空闲时，仿真系统认为员工 A 可以执行活动，而仿真时间处于工作时间表之外时（如周一晚上 10 点），系统认为员工 A 无法执行活动，或者只能加班执行活动。

4) **其他仿真设置**

运行仿真之前，除了要设置仿真运行、设置事务生成器设置和定义工作时间表外，还要根据不同仿真软件的具体要求，设置其他的仿真选项，例如，WorkflowBPR 软件就要求用户选择是否要动画显示仿真过程、动画显示的速度是快还是慢。

2 仿真执行

设置完仿真环境后，用户就可以运行工作流模型的仿真。仿真时，用户可以在同一个仿真场景下运行多个工作流模型，比较多个过程在相同的仿真环境下的性能；也可以在不同的仿真场景下运行同一个工作流模型，考察仿真环境对仿真结果产生的影响。

仿真过程中，仿真系统自动的模拟活动的执行。当且仅当以下三个条件满足时，系统认为活动可以被执行。

1) 分配给活动的资源空闲，而且可用；

2) 活动输入条件满足；

3) 触发活动的系统事件发生。

系统模拟执行活动的过程实际就是累加活动的执行时间、修改相关资源的状态、计算活动成本和记录其他仿真数据的过程。当这一系列数据操作完毕，就可认为一个活动执行完毕。

仿真系统将选择收集仿真过程中的各种数据，以生成仿真报告。

仿真系统模拟活动执行的具体操作对用户来说是不可见的。但是许多软件都提供了动画显示仿真过程的功能，使得用户可以比较清楚直观的看到仿真过程中哪些活动正在被执行，哪些资源处于空闲状态，哪些资源正被占用。

3 仿真分析

仿真过程中，仿真系统将有选择的收集仿真数据，以便仿真结束时生成仿真报告供用户定量地分析工作流模型性能使用。通常，仿真系统收集的仿真数据包括活动的执行时间、资源的使用时间、活动的成本、事务的等待队列等。

1） **时间**

时间 T 是衡量企业经营过程优劣的一个重要指标。总体的经营过程时间（如过程周期时间）反映了企业响应市场需求的能力，时间越短，说明企业能越快地满足顾客的新需求，越有机会获得市场商机。对于服务型行业来说，顾客等待时间短也意味着服务质量高。而考察分类的过程时间（如活动的执行时间和等待时间）还可以分析工作流模型的运行性能。

工作流模型运行的周期时间（事务时间）定义为一项事务开始到事务处理完毕的时间总和，以 10.3.3 中的例 1 为例，从顾客提交订单到订单处理完毕（软件和发票都邮寄给顾客）所经历的时间是该工作流模型运行的周期时间。通过统计过程中每个活动的执行时间可以得到运行的周期时间。

每一个活动处理的时间由工作时间和等待时间两部分组成，如图 10.14 所示。

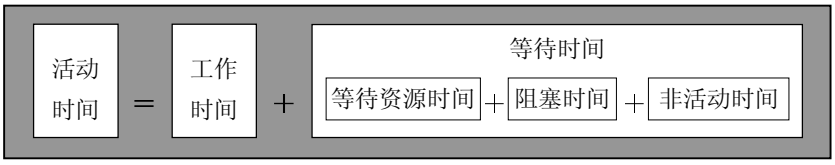


图 10.14 活动处理时间

- 工作时间：指活动执行事务所需的时间。
- 等待时间：指事务已到达活动，但是活动由于种种原因没有处理事务而让事务在队列中等待的时间。根据事务处于等待状态的原因，等待时间又可分为：
  - 等待资源时间：指由于活动执行所需的资源被别的活动占用或损坏、事务等待资源被释放或可用的时间；
  - 阻塞时间：指事务因以下原因不能被活动执行而阻塞在队列中的时间，如活动执行需要满足的批处理条件不成立、正在处理的事务数量已经达到了活动的最大处理能

力等。

– 非活动时间：指资源处于工作时间表之外的时间，如休息时间。

例如，某汽车修理中心的工作时间表如图 10.15 所示。

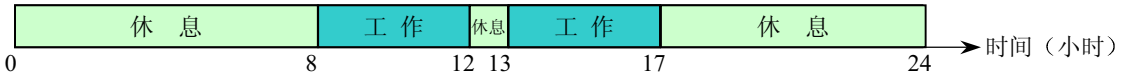


图 10.15 某汽车修理厂工作时间表图例

一辆汽车（事务）下午 4:00 进入修理中心，维修时间需要 2 小时，但是修理人员下午 5 点就下班了，第二天早上 8 点才上班。上班之后，由于修理人员要忙别的事情，直到 9 点才开始继续维修该汽车。因此，该事务的周期时间、工作时间和等待时间如表 10.2 所示。

表 10.2 过程时间统计表－小时

事务数量	周期时间	工作时间	资源等待时间	阻塞时间	非活动时间
1	18.00	2.00	1.00	0.00	15.00

类似的，也可根据资源的状态对事务处理过程中资源的时间进行分类，资源时间由工作时间、空闲时间、无效时间和非活动时间四部分组成，如图 10.16 所示。

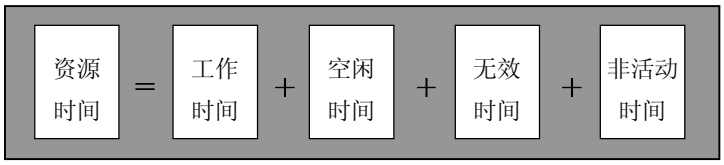


图 10.16 资源时间的组成

一个资源的资源时间在数值上等于从仿真开始到仿真结束的这一段时间，即等于工作流运行的周期时间。其中，

- 工作时间：指资源用于处理事务状态的时间。
- 空闲时间：指资源处于空闲状态（资源可用但目前没有参与处理事务）的时间。
- 无效时间：指指资源因为失效或损坏等原因而不能被使用的时间；
- 非活动时间：指资源因为处于资源工作时间表之外而不能使用的时间。

根据资源时间可以统计资源的利用率。资源利用率是衡量资源使用状况的最常用的指标。

$$\text{资源利用率} = \frac{\text{资源忙时间}}{\text{资源时间} - \text{非活动时间} - \text{无效时间}}$$

例如，假设雇员 A 的工作时间表如图 10.15 所示，对两个工作日内的经营过程进行仿真分析，则资源时间是 48 个小时，其中休息时间（非活动时间）是 32 个小时。这两天中，A

只花了 5 各小时处理事务，其他的时间空闲，则资源 A 的利用率为

$$\frac{5}{48-32-0}=31.25\%$$

2) **成本**

成本 C 是衡量企业经营过程的另一项重要指标。企业的销售收入减去所花费的成本等于企业所获得的收益，在企业销售收入相同的情况下，付出的成本越少获得的利润就越大。

通常，采用基于活动的成本计算方法来统计经营过程的总成本。每个活动成本的构成方式如下：

活动成本＝（标准成本）＋（超时成本）

＝（劳动力成本＋设备成本＋原材料成本＋其他）(在工作时间表之内进行) ＋

（劳动力成本＋设备成本＋原材料成本＋其他）(在工作时间表之外进行)

例如，假设活动 A 是复印一本 500 页的书，需要操作员 1 名，他的标准工资是 10 元/小时，加班工资是 20 元/小时；复印机每复印 1000 页纸的设备折旧费和电费共计 40 元；每 500 张 A4 复印纸的价格是 32 元。操作员的工作时间表与图 10.15 所示的工作时间一样。由于任务紧急，该操作员从下午 4:30 一直工作到 5:30 才把书复印完毕，则该复印活动的

劳动力成本 ＝ （标准工资×正常工作时间）＋（加班工资×加班工作时间）

＝ （10×0.5）＋（20×0.5）

＝ 15 元

设备成本 ＝ 复印一张纸的设备成本×复印纸张数

＝ （40 ÷ 1000）×500

＝ 20 元

原材料成本 ＝ 每张 A4 复印纸成本×复印纸张数

＝ （32 ÷ 500）×500

＝ 32 元

其他 ＝ 0 元

因此，该复印活动的总成本 ＝ 15 ＋ 20 ＋ 32 ＋ 0

＝ 67 元

知道了每一个活动的成本，就可以进一步统计经营过程的成本。根据仿真分析的目的不同，统计经营过程运行成本的方法也不同。经营过程成本的统计方法可以分为三类。

- **按价值类型统计**

经营过程中的活动可以分为增值活动、商业增值活动和非增值活动。按价值类型统计时，总成本由增值活动成本、业务增值活动成本和非增值活动成本三部分组成，如图 10.17 所示。

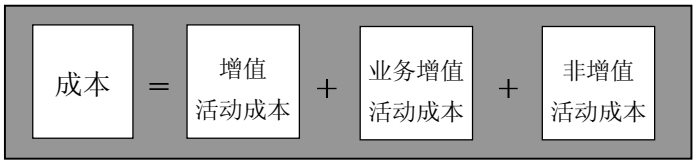


图 10.17 按价值类型统计成本

其中，增值活动是指那些直接为产品的生产或服务的开展创造了价值的活动，例如，将零配件组装成产品就是一个增值活动。业务增值活动是指那些没有直接为产品的生产或服务的开展创造价值、但却是维持正常的经营过程运行所必须的活动。典型的业务增值活动如生产管理活动。而非增值活动是指那些既为产品的生产或服务的开展创造价值，又不是维持正常的经营过程执行所必须的活动，但是由于受到客观条件的限制而必须执行的活动，如要把车间 1 生产出的半成品转移到另外一个车间 2 进行加工，这种半成品的转移活动就是非增值活动。

- **按资源类型统计**

按资源类型统计时，成本由劳动力成本、设备成本、原材料成本和其他可计算成本构成，如图 10.18 所示。

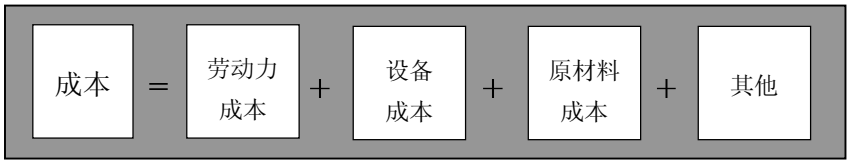


图 10.18 按资源类型统计成本

- **按时间类型统计**

按时间类型统计时，成本由标准成本和超时成本构成。标准成本是指活动在工作时间表之内处理事务所消耗的成本，而超时成本是指活动在工作时间表之外处理事务的成本，如图 10.19 所示。

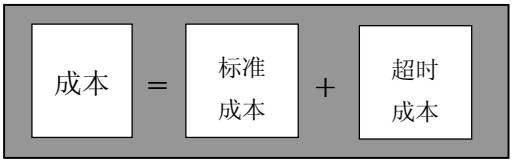


图 10.19 按时间类型统计成本

采用不同方式统计出的成本在数值上一样的，即价值类型成本 = 资源类型成本 = 时间类型成本，但是它们的意义却不同，因此，可以根据不同的仿真目的选择不同的成本统计

方法。例如，当要比较分析过程中增值活动、业务增值活动和非增值活动分别花费的成本，考察非增值活动是否占用了大量的成本、或者决定是否要改进流程设置，减少不必要的商业增值活动和非增值活动时，可以按价值类型统计经营过程的成本。当要了解过程中各类型资源的消耗成本时可按资源类型统计经营过程的成本。当要考察活动的超时成本是否远远大于标准成本，决定是否要加班处理事务时，则可按时间类型统计成本经营过程的成本。

3） **等待队列**

当事务等待处理时，就形成等待队列。统计每个活动的平均等待队列长度可以衡量活动处理事务的能力、或者分析分配给该活动的资源数量是否合理。

仿真系统在收集活动时间、资源时间、活动成本和事务队列长度等仿真数据的基础上生成仿真统计报告，统计报告多以表格、直方图、圆饼图和曲线图的形式出现。很多软件都提供了灵活的报表定置功能，使用户可以根据各自的需求定义表报的格式和内容。通常来说，仿真报告的主要内容包括：

1）资源统计报告

资源统计报告用来跟踪经营过程中资源的使用情况，这里所说的资源包括经营过程中涉及到的人、设备和物料。对于人和设备来说，资源统计报告的内容包括：

- 资源参与了哪些活动的执行；
- 资源的使用时间；
- 资源的空闲时间；
- 资源的平均负荷；
- 资源消耗的成本在其参与活动的总成本中的百分比；

对物料来说，资源统计报告的内容包括：

- 物料参与了哪些活动的执行；
- 物料的使用时间；
- 物料在队列中的平均等待时间；
- 物料的消耗数量及其占总物料量的百分比。

从资源统计报告中，我们可以分析在经营过程中资源什么时候被使用，哪一种资源的使用率最高，哪一些资源负荷太重，哪一些资源比较空闲，哪一些资源是过程执行的瓶颈资源。由此可以确定当前的资源配置是否合理，是否需要增加新的设备或者雇员，或者减少一些使用率很低的设备。

2）过程/活动统计报告

过程/活动统计报告是仿真统计报告中的另一项重要内容，它可以比较全面的评价经营过程的运行性能以及经营过程中活动的执行情况。过程/活动统计报告一般包括以下内容：

- 经营过程的时间，包括过程周期时间、过程中事务的执行时间和等待时间等；
- 经营过程成本，指整个过程的总成本；
- 经营过程的产量；
- 经营过程中已执行活动的个数和执行次数；
- 活动的执行时间，及其占经营过程时间的百分比；
- 活动等待时间，及其占经营过程等待时间的百分比；
- 活动成本，以其占经营过程成本的百分比；
- 活动消耗资源的种类和数量；
- 活动的平均队列长度。

从过程/活动报告中，我们可以分析给定经营过程的时间、成本和产量等运行指标，并可以判断过程中，哪个活动是最费时的，哪个活动的执行最容易受阻，哪个活动的成本最高，哪个活动消耗了最多的资源，哪些活动的资源配置不合理导致事务的等待队列较长，从而确定经营过程中的关键活动，理顺阻碍活动的各项因素，从降低活动成本入手降低整个经营过程的总成本。

4 仿真模型修改

在对多次仿真运行的结果进行分析之后，通常还要修改工作流模型或重新定义仿真环境。这样做主要有两个目的，

1） 使工作流模型的描述更加准确。一开始建立的工作流模型可能存在与实际的经营过程不一致的地方，通过分析仿真结果并将其与已知的经营过程的实际性能相比较，可以分析模型中的哪些环节与实际情况不符合。然后通过反复的修改工作流模型并执行仿真，可以使工作流模型逐步逼近实际的经营过程，描述更加准确。

2） 优化企业经营过程。工作流模型仿真的一个重要目的是比较多种经营过程的设计方案以便从其中选择最佳方案。由于修改工作流模型（包括调整模型的结构和修改模型的参数）或重新定义仿真环境可以模拟实际经营过程或运行环境的变化，因此，修改过的工作流模型和仿真环境可以体现一个新的经营过程的设计方案。通过对修改过的工作流模型进

行仿真和对比分析，就可以有效的定量的评价不同的过程设计方案，从中选择最佳方案，实现企业经营过程的优化。

优化企业的经营过程有多种不同的手段，比较常见的例如：改变事务处理的路径、修改分配给活动资源的数量、改变活动的执行时间、改变组织结构或资源结构、或修改资源的数量等。在工作流仿真环境下研究过程改进的方法是重新定义仿真环境，其最常见的内容有修改事务生成的频率和数量、或修改资源的工作时间表等。

通过多次的修改工作流模型及仿真环境，就可以得到令人满意的工作流模型和优化的企业经营过程。

10.4.4 工作流模型仿真分析实例

不同的市场环境对企业经营过程的评价指标有不同的要求。在产品供不应求的市场环境下，生产效率、生产成本和产品产量是企业最关心的指标。企业希望能够最大限度的提高劳动生产率，降低生产成本，以更低的成本生产出更多的产品来满足市场的需要，并获得最大的利润。随着卖方市场向买方市场的转变，市场竞争的加剧，顾客需求变化频率的加快，使得企业从单纯的注重成本和产量转向注重上市速度、产品质量、产品创新和顾客服务。上市时间（Time）、产品质量（Quality）、过程成本（Cost）、和顾客服务水平（Service）被人们普遍认为是影响企业竞争能力的关键因素。因此，人们也希望从这四个方面来综合的评价企业的经营过程。

由于“质量”和“服务”是两项比较难量化的指标，对它们的评价一般采用基准检查的方法来衡量，也就是以本行业内业绩好的企业为基准进行比较，评价本企业的质量和服务水平。例如，可以从以下几个方面来衡量企业提供服务的水平。

- 是否能够按时完成用户订单，在指定时间将货物运送到指定地点？
- 服务时间是否足够长，使用户可以随时获得企业提供的服务？
- 产品是否有足够长的保修期，一旦产品出现故障，企业能及时提供维修服务？
- 是否能及时的为用户提供备品备件？
- 是否能及时的为用户提供产品信息、技术支持或技术培训？

一般来说，通过工作流模型仿真，人们可以从过程时间、过程成本、资源使用状况和事务等待队列这四个方面来综合的分析和评价企业的经营过程的性能。值得一提的是，这些量化指标之间往往存在着相互影响、相互制约的关系，例如一个制造流程中，增加工人的数量

可以缩短过程的周期时间，但却会导致过程成本的上升。因此，只有全面的考虑这些指标，分析指标之间的相互关系，才能够准确的评价经营过程。

下面，仍以 10.3.3 中软件公司 A 的软件供应及开发票的业务过程为例，采用 iGrafx 软件对它进行仿真分析。对于这样的软件公司来说，如何在接到顾客的订单之后用最短的时间把软件产品和发票送到顾客手中是企业最关心的事情。因此，仿真的主要目的是要分析该业务过程的时间特性，确定哪些因素或环节对业务过程的周期时间有重要影响，寻求缩短业务过程周期时间的途径。

1） 设置仿真环境

假设仿真从周一早晨 8:00 开始运行，仿真长度为 1 个月，仿真运行遵守标准的工作时间表（即每天工作 8 小时，每周工作 5 天，一个月有 22 个工作日）。仿真开始时，顾客生成一个订单，以后平均每 6—12 小时有一个新的顾客订单到达。因此仿真环境的设置如下（iGrafx 软件中，仿真运行场景包括资源的分配，但这一工作是在建模阶段就完成的）。

运行设置如下：

```
Create : Report1                          //生成的仿真报告名为 Report1

Simulation Time : Calendar - Standa:      //按标准的日历执行仿真
Simulation Start Time : 星期一  8:0C      //仿真从周一早上 8 点开始运行
Simulation End :  1 Months                //仿真长度为 1 个月

Active Hours/Day : 8                      //每天工作 8 小时
Active Days/Week : 5                      //每周有 5 个工作日
Active Days/Month : 22                    //每个月有 22 个工作日

Sequence : 1                              //仿真用使用一个随机数发生器

Active Limit : 1000                       //可执行的最大活动数是 1000
```

事务生成器设置如下：

```
Generator1  -  Active                     //事务作用于过程的开始活动
 Start
 Generator Type : Interarrival            //事务的类型是间隔型
 Schedule: Default                        //采用的工作时间表是系统默认的时间表
 Start : Simulation Start                 //开始：仿真开始
 End : Simulation End                     //结束：仿真结束
 Initial Count : 1                        //仿真开始有一个事务
 Subsequent Count : 1                     //以后每次都产生一个事务
 InterarrivalDuration = Between(6,12)Hours //事务的间隔规律是每 6 至 8 小时产生一个事务
```

分配给活动的资源如下：

包装运输部	Count ： 5	//有 5 个员工
	Schedule ： Standard	//员工遵守标准作息时间表
	Hourly Rate ： ￥10.00	//员工的工资是每小时￥10
	Hourly O/T Rate ： ￥20.00	//员工的加班工资是每小时￥20
软件复制部	Count ： 2	//有 2 个员工
	Schedule ： Default	//员工遵守标准作息时间表
	Hourly Rate ： ￥10.00	//员工的工资是每小时￥10
	Hourly O/T Rate ： ￥20.00	//员工的加班工资是每小时￥20
生产管理部	Count ： 2	//有 2 个员工
	Schedule ： Default	//员工遵守标准作息时间表
	Hourly Rate ： ￥10.00	//员工的工资是每小时￥10
	Hourly O/T Rate ： ￥20.00	//员工的加班工资是每小时￥20
销售部	Count ： 3	//有 3 个员工
	Schedule ： Default	//员工遵守标准作息时间表
	Hourly Rate ： ￥10.00	//员工的工资是每小时￥10
	Hourly O/T Rate ： ￥20.00	//员工的加班工资是每小时￥20
信用认证和财务部	Count ： 5	//有 5 个员工
	Schedule ： Default	//员工遵守标准作息时间表
	Hourly Rate ： ￥10.00	//员工的工资是每小时￥10
	Hourly O/T Rate ： ￥20.00	//员工的加班工资是每小时￥20

2） 执行仿真

运行仿真时，用户只要从菜单中或工具条中选择“开始”或“终止”按钮就可以开始或结束一次仿真运行。因此，执行仿真的过程对用户来说是非常容易的。同时，iGrafx 软件提供了动画显示仿真过程的功能，使得用户可以监视仿真的进展状况，了解现在正在处理哪些活动，已经执行完了几个活动。

3） 仿真结果分析

仿真运行结束后，iGrafx 软件可以根据用户的定义生成多种形式的时间、成本、资源、队列的仿真报告。由于本次仿真的目的是分析该工作流模型的时间特性，因此，重点分析如下的仿真时间报告和成本报告。

表 10.3 仿真消耗时间统计表 （单位：天）

30.00

表 10.4 事务时间统计表 单位：天

事务数	平均周期时间	平均工作	平均等待资	平均阻塞	平均非活动

		时间	源时间	时间	时间
36	5.40	1.31	.03	<.01	4.06

表 10.5 活动时间统计表 单位：小时

	总周期时间	事务数	平均周期时间	平均工作时间	平均等待资源时间	平均阻塞时间	平均非活动时间
信用认证和财务部 - 审查顾客信用	1248.00	20	62.40	16.00	.00	.00	46.40
信用认证和财务部 - 寄发票	888.00	25	35.52	8.00	.00	.00	27.52
包装运输部 - 邮寄产品	792.00	25	31.68	8.00	.00	.00	23.68
销售部 - 处理信用问题	624.00	18	34.67	8.00	.00	.00	26.67
包装运输部 - 包装	343.00	19	18.05	3.00	.00	.00	15.05
信用认证和财务部 - 产品已邮寄 ?	323.79	25	12.95	.00	.09	2.66	10.20
销售部 - 接收订单	235.00	22	10.68	2.00	.00	.00	8.68
软件复制部 - 复制软件	123.00	19	6.47	2.00	.00	.00	4.47
信用认证和财务部-准备发票	120.00	26	4.62	2.00	.00	.00	2.62
生产管理部 - 订单进入	88.00	27	3.26	1.00	.00	.00	2.26
信用认证和财务部 -资信优良	86.83	27	3.22	.00	.29	.00	2.93
包装运输部 - 核对订单	44.00	26	1.69	.50	.00	.00	1.19
信用认证和财务部 -收到订单	11.74	22	.53	.08	.36	.00	.09
软件复制部 - 生产调度	5.75	19	.30	.25	.00	.00	.05
销售部 - 取消订单	2.83	11	.26	.17	.00	.00	.09
顾客 - 付款过程	.00	25	.00	.00	.00	.00	.00
信用认证和财务部 - 好?	.00	20	.00	.00	.00	.00	.00
销售部 - 解决 ?	.00	18	.00	.00	.00	.00	.00
生产管理部 - 有 库存?	.00	27	.00	.00	.00	.00	.00
顾客 - 生成订单	.00	22	.00	.00	.00	.00	.00

表 10.6 事务成本统计表 单位：元

事务数	平均成本	平均增值活动成本	平均劳动力成本	平均设备成本	平均其他成本
36	￥292.59	￥292.59	￥292.59	￥.00	￥.00

表 10.7 活动队列统计表 单位：个

	事务数	总等待队列	平均等待队列	最大等待队列
信用认证和财务部 - 审查顾客信用	20	22	6	3
信用认证和财务部 - 寄发票	25	25	3	4
包装运输部 - 邮寄 产品	25	26	3	4
销售部 - 处理信用 问题	18	20	2	2
信用认证和财务部 - 资信 优良	27	4	2	2
信用认证和财务部 - 产品已邮寄 ?	25	16	1	2
生产管理部 - 订单 进入	27	5	1	1
信用认证和财务部 - 准备发票	26	13	1	1
软件复制部 - 复制软件	19	9	1	1
软件复制部 - 生产调度	19	1	1	1
信用认证和财务部 - 收到订单	22	4	1	1
包装运输部 - 包装	19	16	1	1

包装运输部 - 核对 订单	26	4	1	1
销售部 - 接收订单	22	11	1	1
销售部 - 取消订单	11	1	1	1
顾客 - 付款过程	25	0	0	0
信用认证和财务部 - 好?	20	0	0	0
销售部 - 解决 ?	18	0	0	0
顾客 - 生成订单	22	0	0	0
生产管理部 - 有 库存?	27	0	0	0

从以上仿真报表中可以知道，活动“审查顾客信用”是业务过程中比较关键的一个环节，它耗时最长，活动的成本最高，而且活动的平均等待队列也最长，是影响业务过程周期时间的瓶颈活动。所以，为了能够缩短过程的平均周期时间，关键是要寻找有效的途径来改善“审查顾客信用”这一环节。

由于“审查顾客信用”的平均等待队列最长，因此，我们考虑给“信用认证和财务部”增加一名雇员，分析这一措施是否能够有效地缩短平均等待队列和过程的周期时间。仿真场景的改动部分定义如下：

```
包装运输部          Count : 5                    //有 5 个员工
                    Schedule : Standard          //员工遵守标准作息时间表
                    Hourly Rate : ￥10.00        //员工的工资是每小时￥10
                    Hourly O/T Rate : ￥20.0     //员工的加班工资是每小时￥20
软件复制部          Count : 2                    //有 2 个员工
                    Schedule : Default           //员工遵守标准作息时间表
                    Hourly Rate : ￥10.00        //员工的工资是每小时￥10
                    Hourly O/T Rate : ￥20.0     //员工的加班工资是每小时￥20
生产管理部          Count : 2                    //有 2 个员工
                    Schedule : Default           //员工遵守标准作息时间表
                    Hourly Rate : ￥10.00        //员工的工资是每小时￥10
                    Hourly O/T Rate : ￥20.0     //员工的加班工资是每小时￥20
销售部              Count : 3                    //有 3 个员工
                    Schedule : Default           //员工遵守标准作息时间表
                    Hourly Rate : ￥10.00        //员工的工资是每小时￥10
                    Hourly O/T Rate : ￥20.0     //员工的加班工资是每小时￥20
信用认证和 财务部   Count : 6                    //有 6 个员工
                    Schedule : Default           //员工遵守标准作息时间表
                    Hourly Rate : ￥10.00        //员工的工资是每小时￥10
                    Hourly O/T Rate : ￥20.0     //员工的加班工资是每小时￥20
```

在新的仿真场景下执行仿真，从仿真报告表 10.8 和表 10.9 中，我们吃惊的发现，在“信用认证和财务部”增加了一名雇员之后，事务的等待队列和业务过程的周期时间非但没有缩短，反而增加了。这表明采用增加员工的办法不能实现缩短事务等待队列的目的。

表 10.8 事务时间统计表 单位： 天

事务数	平均周期时间	平均工作时间	平均等待资源时间	平均阻塞时间	平均非活动时间
40	5.60	1.35	.01	<.01	4.24

表 10.9 活动等待队列统计表 单位：个

	事务数	总等待	平均等待队	最大等待

		队列	列	队列
信用认证和财务部 - 审查顾客信用	21	21	7	4
信用认证和财务部 - 寄发票	32	33	3	4
包装运输部 - 邮寄 产品	32	33	3	4
销售部 - 处理信用 问题	20	21	2	2
信用认证和财务部 - 产品已邮寄 ?	33	15	1	2
信用认证和财务部 - 资信优良	33	4	1	1
生产管理部 - 订单进入	33	9	1	1
信用认证和财务部 - 准备发票	33	16	1	1
软件复制部 - 复制软件	20	11	1	1
信用认证和财务部 - 收到订单	21	4	1	1
包装运输部 - 包装	20	14	1	1
包装运输部 - 核对订单	33	4	1	1
销售部 - 取消订单	8	1	1	1
销售部 - 接收订单	21	13	1	1
顾客 - 付款过程	32	0	0	0
生产管理部 - 有库存?	33	0	0	0
软件复制部 - 生产调度	20	0	0	0
信用认证和财务部 - 好?	21	0	0	0
销售部 - 解决 ?	20	0	0	0
顾客 - 生成订单	21	0	0	0

重新审查业务过程，发现在现有的业务过程中，无论是新顾客还是老顾客都要经过信用认证。这一过程是不够合理的，因为老顾客在以前的业务活动中已经通过了信用认证，如果对其进行再次认证就是无谓的重复性劳动。在这种情况下，决定修改信用认证环节：对于老顾客，认为其资信良好而不再对其进行认证；对于新顾客，则按正常的程序对其进行信用认证。工作流模型的修改部分如图 10.20 所示。

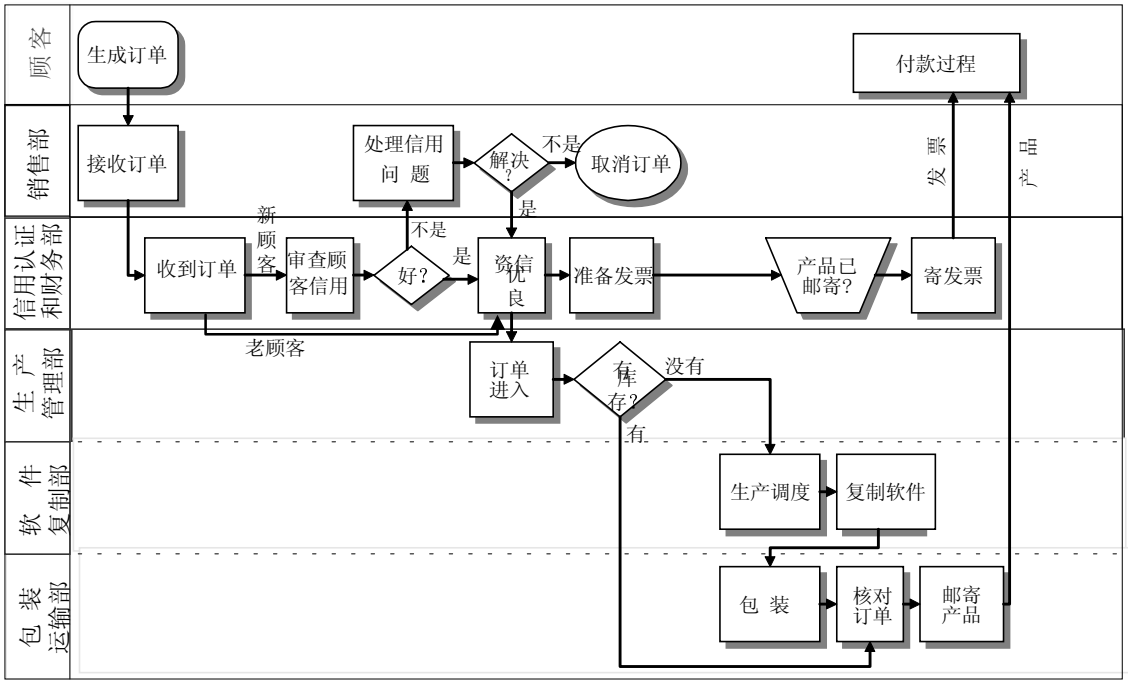


图 10.20 修改后的过程模型

假设一份订单来自新顾客的概率是 30%，来自老顾客的概率是 70%，对新的工作流模型进行仿真（不修改仿真场景），得到如表 10.10、表 10.11 和表 10.12 所示的仿真报表。

表 10.10 事务时间统计表 单位：天

事务数	平均周期时间	平均工作时间	平均等待资源时间	平均阻塞时间	平均非活动时间
23	4.13	.97	.00	.00	3.16

表 10.11 事务成本统计表 单位：元

事务数	平均成本	平均增值活动成本	平均劳动力成本	平均设备成本	平均其他成本
23	￥266.78	￥266.78	￥266.78	￥.00	￥.00

表 10.12 活动等待队列统计表 单位：个

	事务数	总等待队列	平均等待队列	最大等待队列
信用认证和财务部 - 审查顾客信用	7	8	5	3
信用认证和财务部 - 寄发票	19	20	3	4
包装运输部 - 邮寄 产品	19	20	3	4
销售部 - 处理信用 问题	6	7	2	2
信用认证和财务部 - 准备发票	20	11	1	3
信用认证和财务部 - 产品已邮寄 ?	20	12	1	2
包装运输部 - 包装	12	8	1	2
生产管理部 - 订单 进入	20	4	1	1
软件复制部 - 复制软件	12	8	1	1
软件复制部 - 生产调度	12	1	1	1
包装运输部 - 核对 订单	20	2	1	1
销售部 - 接收订单	20	9	1	1
信用认证和财务部 - 资信优良	21	0	0	0
顾客 - 生成订单	20	0	0	0
信用认证和财务部 - 收到订单	20	0	0	0
生产管理部 - 有库存?	20	0	0	0
顾客 - 付款过程	19	0	0	0
信用认证和财务部 - 好?	7	0	0	0
销售部 - 解决 ?	6	0	0	0
销售部 - 取消订单	4	0	0	0

从仿真报表中，我们不难看出，在修改了企业 A 业务过程中的信用认证环节之后，过程的周期时间显著缩短，从以前的 5.40 天缩短为 4.13 天。事务的平均成本也略有下降，从以前的 292.59 元降到改变后的 266.78 元。活动“审查顾客信用”处理的事务数量明显减少，从以前的 40 个事务中需要审查 20 次顾客信用降到改变后的 23 个事务中需要审查 7 次顾客信用；而且它的平均等待队列也有所下降，从以前的 6 个降到改变后的 5 个。由此可见，修改企业 A 业务过程中的信用认证环节使业务过程的性能有了较大的改善。

10.5 基于工作流管理的经营过程重组实施

在建立了工作流模型并对模型进行分析和优化之后，就可以根据工作流模型实施企业经营过程的改造，并可以据此建立相应的工作流管理系统。

从前面的介绍中，我们已经知道工作流管理技术具有良好的可实施性。因为工作流模型的建立方式实现了过程逻辑与应用逻辑的分离，它可以在不修改具体功能模块实现方式的情况下，通过修改过程逻辑（工作流模型）来改进系统性能，实现对经营过程的集成管理。这种方式在企业应用时具有显著的优点：它能够有效地把人、信息和应用工具合理地组织在一起，同时又能够提高软件的重用率，具有实施灵活的特点。因此，工作流管理技术经常与企业经营过程重组的改造实施联系在一起。

工作流管理系统中，工作流机根据工作流模型的定义创建过程实例，并调度和监控过程中每个活动的执行，在需要的场合触发软件应用或者完成计算机应用与操作人员的交互，从而实现对过程的有效的管理和控制。由于这一过程需要集成具体的业务应用软件和操作人员的用户界面，因此在企业或部门的经营过程中建立工作流管理系统的过程是是一个业务应用软件系统的集成与实施过程。

经营过程以及组成活动的复杂程度的不同，工作流管理系统覆盖经营过程的范围（部门、全企业、企业间）也不同，工作流管理系统可以采取许多种实施方式。但是无论采用哪种方式，都需要有相应的网络技术、通讯技术、数据库技术、计算机支持协同工作技术、和人机接口等多种支撑技术和管理技术的支持。

1．实施原则

在企业实施工作流管理系统进行过程改造是一项非常复杂、高度集成的工程，尤其是我国企业目前的普遍状况是管理水平比较落后、车间自动化水平低、计算机应用技术还未普及、企业的资金有限，因此，在我国企业实施工作流管理系统应该遵循如下的原则：

1） 效益驱动的原则

效益是企业的生命线。企业实施工作流管理系统时应该首先选择对企业有重要影响的关键业务过程进行改造，使项目有较好的投入/产出值，从而形成良好的经济效益和资金的循环。

2） 有限目标、有限投资、分期实施和逐步发展的原则

在企业实施工作流管理系统时，应该充分考虑我国企业的实际情况，考虑人员、技术储

备、管理水平和资金条件的限制，在总体设计中确定合理的、可实现的和可持续发展的系统目标。根据企业可能作出的投资规模，从企业当前和长远的需求出发，制定合适的目标，分期实施，逐步发展，便于今后在此基础上持续发展。

3） 信息集成、过程集成和人的集成并重的原则

企业的经营过程的改造是整体的、全方位的改造，其意义决不仅仅限于技术革新或过程的自动化。因此，在实施工作流管理系统时应从全局出发，综合考虑企业的物流、信息流和工作流的合理组织和集成，重视人员培训，最终实现企业全面集成。

4） 建立开放的分布的企业计算机基础设施的原则

计算机基础设施是实施工作流管理系统的重要前提。对企业而言，信息基础设施主要包括计算机网络和数据库管理系统。在建立信息基础设施的时候，既要考虑企业的当前需求，又要考虑企业未来发展和技术发展的需要，为今后的发展留有一定的发展空间。

5） 与企业的机制改革相结合的原则

企业的运行机制与工作流管理系统能否高效的运转有重要关系。在建立工作流管理系统的同时也应该着手改革企业的运行机制，使运行机制与新的业务流程相匹配，获得最佳的实施效果。

2．实施内容

在企业建立工作流管理系统以实施过程改造的过程包括以下内容，

1） 建立计算机支撑环境和安装测试各种硬件设备

建立计算机支撑环境的包括：

- 铺设计算机网络，安装相应的网络设备和网络软件系统，并对网络性能进行测试。
- 根据信息视图建立相应的数据库系统。

2） 实施应用系统

工作流管理系统的核心部件－－工作流管理软件既可以自行开发，也可以选购成熟的商品化软件。针对企业的具体情况自行开发的工作流管理软件具有适应性好、针对性强等优点，但是开发周期比较长，费用高。因此，除非企业的业务过程非常特殊，一般应尽量选购成熟的商品化软件，在此基础上对软件进行二次开发，使软件能够满足企业的具体需要。同时还要开发工作流管理软件与企业中已有的及其他应用软件的接口。

3） 企业内部调整

人是企业中的生产要素之一，也是企业中最活跃、最宝贵的因素。由于工作流管理系统的实施会改变人的工作方式和传统的管理观念，还有可能影响个人的利益，往往使人们对工作流管理系统的实施产生误解和抵触情绪，因此，实施过程中要充分重视对人员的培训。通过培训，一方面可以提高人的思想水平，使他们能够接受先进的管理思想，另一方面也可以提高人的技术水平，使他们能够掌握先进的技术，严格按照操作规范进行操作。

除了进行人员培训之外，企业还要对组织结构和运行机制进行改革，建立相应的组织机构和企业管理制度，保证改进后的经营过程能够发挥更好的作用。

第 11 章 工作流在 CIMS 中的应用

11.1 CIMS的基本概念

计算机集成制造(Computer Integrated Manufacturing , CIM)是美国 Joseph Harrington 博士于 1974 年针对企业所面临的激烈市场竞争形势而提出来的组织企业生产的一种哲理。CIM 的一个比较科学的定义是“CIM 是信息技术和生产技术的综合应用，旨在提高制造业企业的生产率和响应能力，因此，企业的所有功能、信息、组织管理都是一个集成起来的整体的各个部分”。CIM 的通俗含义是用计算机通过信息集成实现企业的现代化生产经营，求得企业的总体效益。按照 CIM 的思想，企业的各个生产环节是不可分割的，需要统一考虑。

CIM 概念的出现得到了世界各国政府，研究所和企业界的广泛重视，其发展十分迅速，并已经在企业应用中取得了显著的成果。其应用范围也从制造业扩展到电子、化工、钢铁、制药、航空航天等行业。CIMS（计算机集成制造系统）是 CIM 思想的实现。根据企业的具体情况不同，CIM 的哲理有各种实现方法，这些实现方法就是 CIMS。

经过 863 计划 CIMS 主题十多年的技术攻关与企业推广应用，CIMS 的研究和应用在我国取得了显著的成果，在国际上也有了较大的影响。标志着我国 CIMS 研究与应用的具有代表性的成果有：清华大学国家 CIMS 工程技术研究中心获得美国制造工程师协会颁发的大学领先奖、北京第一机床厂获得美国制造工程师协会颁发的工业领先奖、近 200 家企业已经开始进行 CIMS 应用工程的实施工作，许多企业已经通过信息集成取得了显著的效益。

CIMS 通过网络和数据库把企业的管理、生产控制、工程设计等子系统集成起来，使企业能具有快速的市场响应能力（T)、高产品质量（Q)、低生产成本（C）和良好的服务（S)，以适应剧烈的市场竞争需要。CIMS 在原有的管理、工程、自动化等技术的基础之上把经营系统、人的系统和技术系统集成。 企业只有通过集成，才能使企业各部门的活动从总体上协调，产生总体效益。因此企业实施 CIMS 就是在 CIM 思想指导下，通过网络和数据库系统的支持，实现 CAD/CAPP/CAM, MIS（管理信息系统），MAS（制造自动化系统）各分系统内部和各系统之间的集成，使各分系统协调运行，达到全局优化，最大限度地达到整体效益。

实施 CIMS 的关键在于集成。企业实施 CIMS 的目的在于取得总体效益，而企业能否获得最大的效益又取决于企业各种功能的协调。一般来说，企业集成的程度越高,这些功能就

越协调，竞争取胜的机会也就越大。因为只有各种功能有机地集成在一起才可能共享信息，才能在较短的时间里作出高质量的经营决策， 才能提高产品的质量，降低成本，缩短交货期。单纯地使用计算机，提高自动化程度而不考虑各种功能的集成，则不可能使企业整体优化，也不可能有效地提高企业对市场的快速响应能力。只有集成才能使"正确的信息在正确的时刻以正确的方式传到正确的地方"。因此集成是构成整体， 构成系统的主要途径， 是导致整个企业成功的关键因素。所以说 CIMS 的核心在于集成。

CIMS 的集成可以分成局部集成和全局集成。局部集成是指企业一个子系统或企业一个部门的集成。如 CAD/CAPP/CAM 系统的集成，车间控制、计划、调度和制造设备的集成。全局集成指整个企业所有子系统的集成。按照集成所要实现的功能可以分成信息集成、过程集成和企业间集成。信息集成指在全企业范围内(设计部门，管理部门，制造部门等)实现信息共享，资源共享，信息集成的基础首先是物理设备(计算机网络，数据库，制造设备)的集成。

市场竞争的日益激烈，要求企业不断提高它的 T、Q、C、S 水平，制造业的生产模式也在从传统的大批量生产向多品种、小批量的生产方式转化。单纯的信息集成已经不能满足企业为适应市场竞争的需求。从并行工程、经营过程重组、敏捷制造、虚拟制造等对于企业 CIMS 应用支持系统的需求中可以清楚的认识到企业的 CIMS 应用集成必须由信息集成向过程集成和企业间集成方向发展。

所谓过程集成是指利用计算机软件集成支持工具（如工作流管理系统、集成平台、集成框架等）高效、实时的实现 CIMS 应用间的数据、资源的共享和应用间的协同工作，将一个个孤立的应用过程集成起来形成一个协调的企业 CIMS 运行系统。实现过程集成后，我们就可以方便的协调各种企业功能，把人和资源、资金及应用合理的组织在一起，从而获得最佳的运行效益。这样，企业就使客户处于主动地位，可以按照客户的需求制订过程，再按照过程来组织功能和工作组。另外，过程集成实现了应用逻辑与过程逻辑的分离及过程建模与具体数据、功能的分离，这样就可以在不修改具体功能的情况下，通过修改过程模型来完成系统功能的改变，从而大大的提高企业灵活性和反应能力。

由此，企业的划分不再是传统的“面向职能”的陈旧模式，而改变为“面向客户”、“面向市场”的新的经营结构。在这一转变过程中需要对企业各种过程活动进行确定、描述、分析和再设计，从而有效的建立起相匹配的新的企业运行机制和组织结构。同时，过程集成涉及到过程的各项性能指标的控制、管理和优化，只有有效的实现企业过程的各项性能指标的跟踪、优化和控制，才能有效的改善和重组企业经营过程，提高企业的竞争能力。

因此，实现过程集成离不开企业建模和对过程的仿真及分析技术的支持。

11.2 企业实施CIMS对集成支持系统的需求

企业的信息系统是非常复杂的系统，企业 CIMS 的应用环境具有以下的特点：1）分布性：企业地理位置的分散和部门的划分导致企业的应用系统必然是在网络环境下的分布系统；2）异构性：由于不同部门（管理、设计、制造）的应用需求不同，相应的硬软件供应商不同，因此企业应用系统一般都是运行在由不同硬件平台、操作系统和异构数据库管理系统组成的异构环境上；3）实时性：大部分企业的信息系统在运行过程中有较高的实时性要求，如过程工业中生产信息的收集、制造设备的控制信息和调度信息的发送等；4)自治性：在整个企业实现集成运行的大前提下，不同领域和不同部门的应用系统的运行还应该具有良好的自治性，这样一是可以充分发挥部门的积极性，二是可以避免某个应用的运行错误造成整个系统不能够运行。

CIMS 的实施是为了更好的实现企业的经营目标，因此 CIMS 的实施和运行应该满足企业的几个基本的需求：1）信息集成：这是企业实施 CIMS 的基本需求；2）过程集成：这是实施并行过程、企业经营过程重组、敏捷制造等的需求；3）现有应用的集成：保护企业过去在信息系统上投资的有效措施之一就是能够实现已有应用的集成和封装；4）灵活性：企业的经营过程重组和并行工程的概念和方法都要求企业能够快速的对其生产、经营、设计过程进行灵活的快速重组以满足市场竞争的需求，因此作为支持企业运行的信息系统就要能够实现灵活的、快速的重组，而且能够方便的实现系统的维护和升级；5）可靠性：对于企业这样一个实际的生产经营系统，其信息系统运行的高可靠性具有重要的意义，可靠性的要求一方面反映在不出现错误或少出现错误，另外一方面的要求是在系统出现故障的情况下能够保证系统中重要的信息不丢失，并且能够快速进行故障诊断和系统恢复；6）安全性和保密要求：在当今网络环境下，保证企业的信息系统不被非法侵入和破坏已经成为许多企业十分关心的问题。工作流管理技术为实施 CIMS 应用集成提供了良好的基础。

11.3 基于工作流的CIMS应用集成

虽然工作流管理系统为 CIMS 应用集成提供了良好的基础，但是为了实现对企业实施 CIMS 的支持，目前市场上现有的工作流管理系统在功能还远远不够。这主要反映在：1）现有的工作流管理系统基本上是一个任务管理系统，它主要实现按照一定的流程对任务进行

管理和活动间控制流的导航，对于 CIMS 中需要实现的信息集成和数据管理的支持能力弱，尤其是缺乏信息集成机制和企业信息模型管理功能；2)目前的工作流管理系统在支持异构分布应用上能力不足，尤其是应用集成和应用封装能力不足； 3）对于企业 CIMS 环境下分布应用的管理和监控能力不足，目前的工作流管理系统在企业组织模型上提供了一定的建模和管理能力，但是在资源模型管理能力上与企业实际应用需求差别较大，对于不是由工作流管理系统直接启动的应用则没有任何管理能力（缺乏用户管理、软件管理、配置管理、权限管理等功能）。

为了解决以上指出的工作流管理系统在支持 CIMS 应用集成方面存在的不足，需要从工作流模型的定义到模型的执行、分布式工作流机的实现等方面扩展现有工作流管理系统的功能，在模型定义上，本书作者提出了一种扩展的工作流模型如图 11.1 所示。

扩展的的工作流模型由 4 部分组成，它们分别是过程模型、组织模型、资源模型以及工作流相关数据。有关该模型的详细介绍见本书 9.2 节。

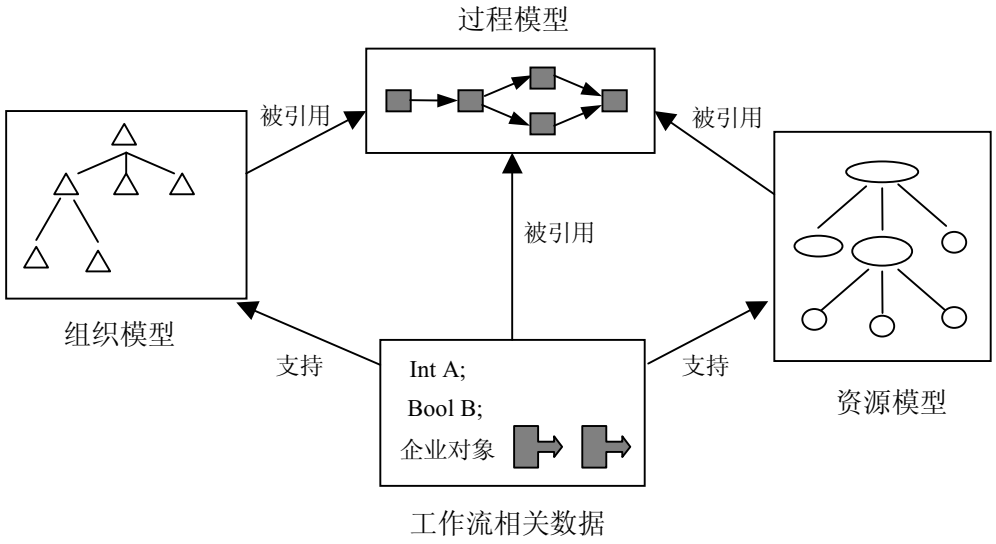


图 11.1 扩展的工作流模型的组成

在以上扩展的工作流模型的基础上和CORBA软件的支持下，采用分布式的工作流执行系统可以良好的实现应用集成。由于采用的是分布工作流机实现方式，在完成工作流过程的定义后，需要按照每个工作流机管理的范围对过程模型进行合理分割，然后分别把各个部分传递给相关的执行工作流机，使每个执行工作流机都保留一部分自己管辖范围内的工作流活动定义，使每个执行工作流机自身都具备自主执行的能力。

企业可以根据自身的实际情况，按照部门和应用领域来配置执行工作流机，这种配置在建模时体现在对每一个基本活动领域属性的定义上，建模时定义了不同的领域属性，在运行时就意味着将由不同的执行工作流机来负责管理、执行。系统中还设置一个主控工作

流机，记录各个执行工作流机的配置情况，并对执行工作流机的状态进行监控，同时负责生成工作流过程实例，并执行管理功能。

应用的集成是一个十分复杂的问题，我们以下从 2 个角度来探讨应用集成的机制问题：从用户角度和工作流执行的角度。从工作流执行角度，可以采用如下几种方式向用户提供不同层次上的应用集成功能：

1) **激活式：**这是最简单意义上的集成，用户给出应用的可执行文件名（包括指定路径、输入文件名、输出文件名等），由工作流机直接调用执行。当应用被激活以后，工作流机就不再对它进行控制了。这种方式无需对应用程序进行包装（Wrap），只是一种命令行的执行方式；

2) **API 式：**这种方式要求被集成的应用提供一定形式的、可被调用的 API 函数，集成方的程序通过调用这些 API 函数来控制应用程序的启动、相关的执行操作、以及最后的退出。比如，利用 Microsoft 提出的 MAPI 来实现邮件的收发操作；

3) **控件式（OLE、ActiveX 方式）：**这种方式的集成主要适于常见的桌面应用（Windows 下）间的互操作，比如 Office 内部组件间的嵌入。这种方式实现起来比较灵活，可以利用已有的控件，也可以自己用不同的方法来定制，如 VB、VC 等。这也是一种对象封装的方法，继承了面向对象的许多优点；

4) **包装式（CORBA 方式）：**这种方式是把要集成的应用封装成 CORBA 对象，集成方的程序通过调用封装后的 CORBA 对象所提供的方法对应用程序的有关操作进行控制。这种方式将继承 CORBA 的许多优点，如语言无关性等，但其实现过程也最为复杂，需要第三方 CORBA 产品的支持。

图 11.2 给出了 4 种集成方式各自的调用层次结构：

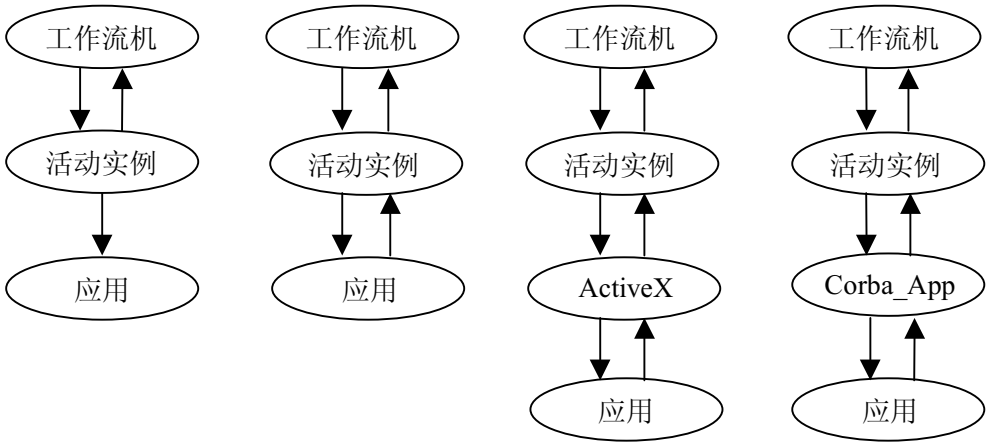


图 11.2 应用集成机制

从用户的角度看，应用集成可以在工作流管理系统提供动态创建工作流模型功能的基础上采用以下几种方式完成：

1) **直接定义：**用户可以通过工作流建模工具定义一个过程模型，在过程模型中确定应用的集成和控制逻辑，然后提交工作流机执行；
2) **嵌入式：**用户定义一个过程模型，并将这个过程作为一个执行函数嵌入某个应用中，在应用执行到这个函数时，工作流机自动完成这个过程模型的执行，实现嵌入式的过程集成；
3) **对话式：**由用户定义一个宏过程，在宏过程的每个步骤中都给出一组可供选择的应用，用户控制整个宏过程的执行，并在每个步骤中选择一个或多个应用提交工作流机去执行，并返回执行的结果。如果不是本地工作流机管理的应用，则通过由本地工作流机负责向其它工作流机提交请求的方式完成应用的执行。

11.4 基于工作流的集成化企业建模方法

企业是一个非常复杂的社会、经济、物理系统。企业实施 CIMS 更是一个复杂的系统工程。为了能够成功的实施 CIMS，除了需要实施者有良好的理论、技术和丰富的实施经验，还需要有先进实用的 CIMS 实施方法论的指导和工具系统的支持。随着 CIMS 理论和实践的发展，针对 CIMS 工程的实施，国内外学者提出了多种系统建模、设计和分析方法论，如 CIMOSA[6]、PERA[60]、ARIS[61]、DEM[62]和 IDEF[8]方法等，这些方法各有特色，也存在不足，它们从不同的角度和出发点提出了对企业这个复杂系统的理解，并给出了描述企业的方法，同时也开发了相应的工具系统。

在当今世界市场正在发生持续快速变化的形势下，企业面临的竞争日益激烈，敏捷制造战略受到越来越多的企业重视，被誉为 21 世纪企业赢得竞争的主要手段。 在这个情况下，对企业建模理论和方法在适应性、集成化、全生命周期、性能优化、一体化建模与实施等方面提出了更高的要求，目前的企业建模方法存在的问题也日益明显，这些问题集中反映在以下几个方面：

1) **现有的建模方法的 3 阶段的生命周期（需求定义、设计、实施）还不能够满足企业持续发展的需要：**在当今持续快速变化的市场竞争环境下，企业的经营过程需要进行不断的重组，生产过程需求进行不断的重构，企业的产品需要进行不断的创新，相应的支持企业实现这些战略目标的企业信息系统就需要不断的进行改进和重组。目前的企业建模的

3 阶段方法仅完成了对企业实施一次信息系统的过程，它不能实现对于企业不断改进和维护信息系统的过程；

2) **现有企业建模方法在集成方面存在严重的不足：**这个不足反映在 2 个方面，第一个方面是大多数企业建模理论和方法采用了多视图建模的方法来描述企业的不同视图侧面，没有一个良好的方法来实现这些不同视图模型之间的集成，虽然 ARIS 方法在这个方面取得了一定的进展，但是由于它的几个视图是独立发展的，用以实现这些独立发展的视图之间集成的控制视图并没有能够解决模型的一致性问题；第二个方面的集成问题是目前的企业建模工具与应用实施工具之间存在严重的脱节，建模理论和方法取得的结果不能直接转化为可实施的系统，导致建模理论的许多成果对于实际企业实施的指导作用没有得到充分的发挥；

3) **现有企业建模工具的适应性和柔性上不足：**现有的企业建模工具基本上是一个结构与功能比较固定的软件系统，在软件的设计与组织上没有使用软件组件的概念和方法，因此所开发的工具系统在适应性和柔性上存在不足。这些不足表现在难以扩展建模工具本身的功能，也难于实现与其它应用系统（如仿真系统）的集成，这些困难在很大程度上制约了建模方法与建模工具的发展与应用效果；

4) **现有建模方法缺乏有效的模型性能评价指标体系：**现有建模方法的模型仿真优化功能较弱，迫切需要建立实用的模型性能评价指标体系，在此基础上进行模型的仿真分析与优化。这个工作在当前快速多变的市场环境下显得尤为重要。在企业的生产经营过程进行重组前，对企业重组的不同方案给出量化指标对于企业进行功能、组织和过程的优化是非常有帮助的，而且量化的指标也企业领导进行决策的重要依据。

我国推广企业 CIMS 应用实施的工作也已经有了一定的规模，取得了显著的效益，并且在实施 CIMS 的过程中积累了相当丰富的经验。但是至今我国企业在实施 CIMS 的工作中还缺乏一个适合的方法论来指导 CIMS 工程的实施，我们还没有一套自己的企业建模和分析工具，企业建模的理论研究和系统实施之间存在相当大的差距，缺少有效的建模和分析工具能够将企业建模理论转换为可操作可利用的企业模型，指导和保证 CIMS 工程的快速实施。因此，为了在更大的范围和更深的层次上推广 CIMS，迫切需要将积累了的丰富经验加以总结，形成具有指导意义的企业建模与分析方法，并提供相应的符合中国企业特点的软件工具，给企业建模和实现基于模型的企业分析提供有效的可操作工具，为企业 CIMS 的应用提供有力的支持。本书作者在此基础上，提出了一套集成化的企业建模方法[57-59]，并给出了其系统体系结构以及基于 CORBA 的企业建模与优化工具实施方法。

11.4.1 集成化企业建模系统体系结构与建模方法

在国内外企业建模理论、方法与工具研究开发成果的基础上，针对以往企业建模方法在视图集成、设计模型和系统实施模型映射等方面存在的不足，结合当今企业对建模方法的需求，本书提出了一套集成化的企业建模方法。提出的建模系统体系结构由生命周期维、视图模型维、通用性层次维组成一个三维的立方体结构（图 11.3），体系结构的每个侧面描述企业建模关心的不同阶段、不同视图和不同的建模构件的通用性程度。虽然本文提出的系统体系结构在很大程度上借鉴了 CIMOSA 的思想，但是本文引入的 4 阶段建模方法及以工作流（过程模型）为核心的建模思想使得本文的方法与 CIMOSA 方法又有了本质上的区别。它更能够满足当今企业对建模方法的需求。以下将就企业建模系统体系结构的各个方面进行说明。

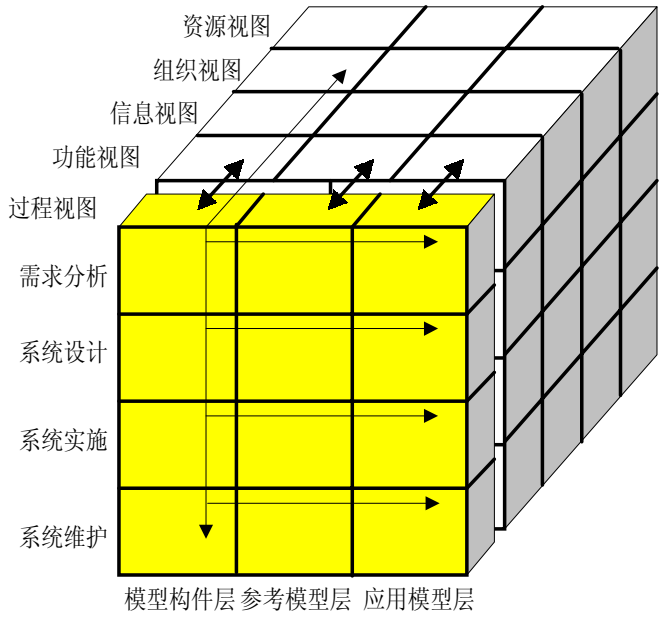


图 11.3 企业建模系统体系结构

1. 生命周期维

在传统的 3 阶段生命周期的基础上，本文提出了从企业需求分析、系统设计、系统实施到运行维护 4 阶段的建模方法。通过引入运行维护阶段，将传统的 3 阶段生命周期建模方法仅支持一次建模—实施过程的开环方式扩展成一个闭环的建模—实施—建模的循环过程，从而将企业建模工作从原来主要在理论和方法上的指导作用延伸为支持企业实现敏捷制造战略、能够快速进行重组以支持企业迅速响应市场的使能系统的重要组成部分。引入运行维护阶段的另外一个重要的原因是企业实施信息系统的实际需求，通过与企业信息系

统实施公司、企业运行维护人员的交流，并通过对当前企业实施信息系统存在的问题的分析研究，我们认识到企业信息系统建立仅是信息系统工作的一小部分工作，更长期和更艰巨的工作是信息系统的维护与升级工作。从目前企业应用软件的发展趋势也可以看出运行维护工作的重要性，在 99 年德国汉诺威 CeBit 计算机博览会上，有许多厂商推出了系统文档维护工具软件，包括 SAP 等大公司也宣称它们的软件具有非常好的可维护性。在此基础上，我们提出将企业建模与实施的 3 阶段生命周期扩展到包括系统维护阶段的 4 阶段生命周期。并且通过这个扩展使整个企业的建模与 CIMS 实施变成一个不断循环，不断上升的过程。系统运行维护阶段工作的结果将是下一阶段企业实施新系统的基础。

以下介绍生命周期中各阶段的研究重点和不同阶段之间的模型映射方法，图 11.4 给出了展开的生命周期维的结构以及不同建模阶段之间的模型映射关系。

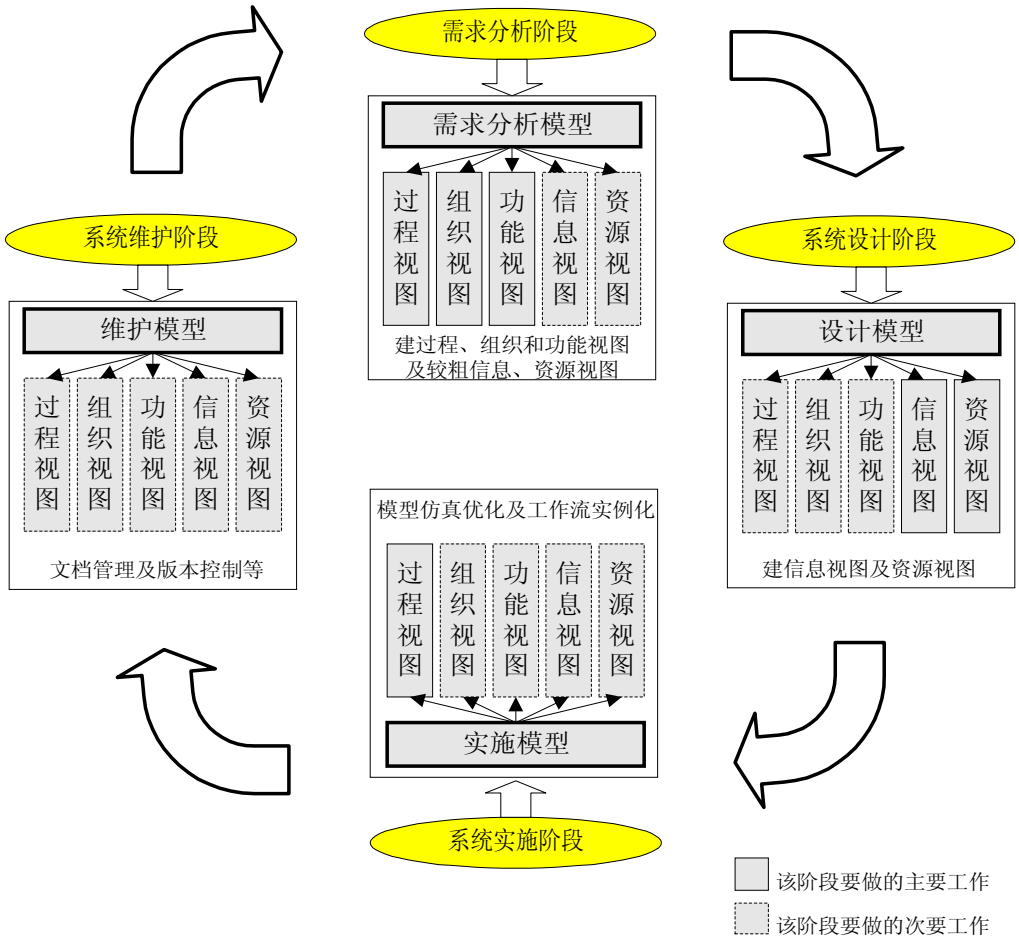


图 11.4 生命周期维结构

由图 11.4 可以看出，整个企业建模实施的生命周期构成一个闭环，每个阶段的结果（输出）是下一个阶段的输入，上一个生命周期的运行维护阶段得到的结果（输出）是下

一个生命周期需求分析阶段的输入，这个不断循环的生命周期反映了企业的 CIMS 系统不断改进和扩展的过程。生命周期各个阶段完成的工作和得到的结果如下:

(1) **需求分析阶段**：在业务调查、现状分析、需求定义的基础上建立需求分析模型，该模型中主要完成过程视图、组织视图和功能视图模型，在过程模型中应包括产品开发设计过程模型的建立，另外在本阶段还应初步建立信息视图和资源视图模型；

(2) **系统设计阶段**：在需求分析阶段所建立的模型基础上，重点完善信息视图及资源视图模型；

(3) **系统实施**（模型仿真及实例化）：在过程视图模型的基础上建立资源视图模型，通过集成的仿真优化分析工具对企业模型进行优化分析，对于经过优化后得到的模型可以利用工作流机进行实例化，从而实现将优化后得到的模型投入实际运行，完成建模过程到实施系统的映射，解决目前企业建模方法与实施系统之间存在的脱节问题；

(4) **运行维护阶段**：对于投入运行的 CIMS 系统进行运行维护，通过文档管理、版本控制等方法实现对于运行系统的有效管理和监控，并通过集成需求管理软件工具来对运行过程中企业不断提出的新的需求进行记录和管理，所积累的需求和文档是下一个生命周期的输入。

2. 视图模型维

本节介绍的集成化企业建模方法采用以过程视图模型（工作流模型）为核心，其它视图（功能视图、信息视图、组织视图、资源视图）模型为辅助视图来实现集成化建模。不同的视图模型之间构成关联和引用的关系。不同视图模型的创建采用逐步建立和完善的方式进行，并通过过程视图作为关键的控制了维护模型之间的一致性问题。视图模型采用软件构件的开发与集成方式，目的是形成具有柔性的动态企业模型。图 11.5 给出了视图模型之间的关系。

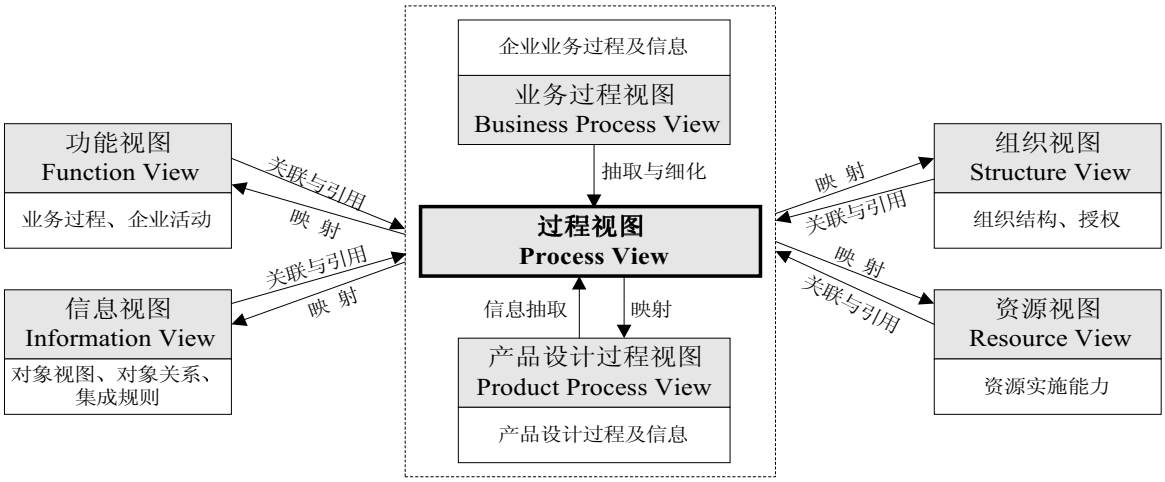


图 11.5 视图模型之间的关系

3）通用性层次维

进行高效快速企业建模的一个重要的有效方法是构建不同建模阶段、不同建模视图的基本构件，并建立基本构件模型库，并以不同的行业为背景建立企业参考模型，从而形成特定的企业模型。图 11.6 给出了模型从通用到专用的发展过程。

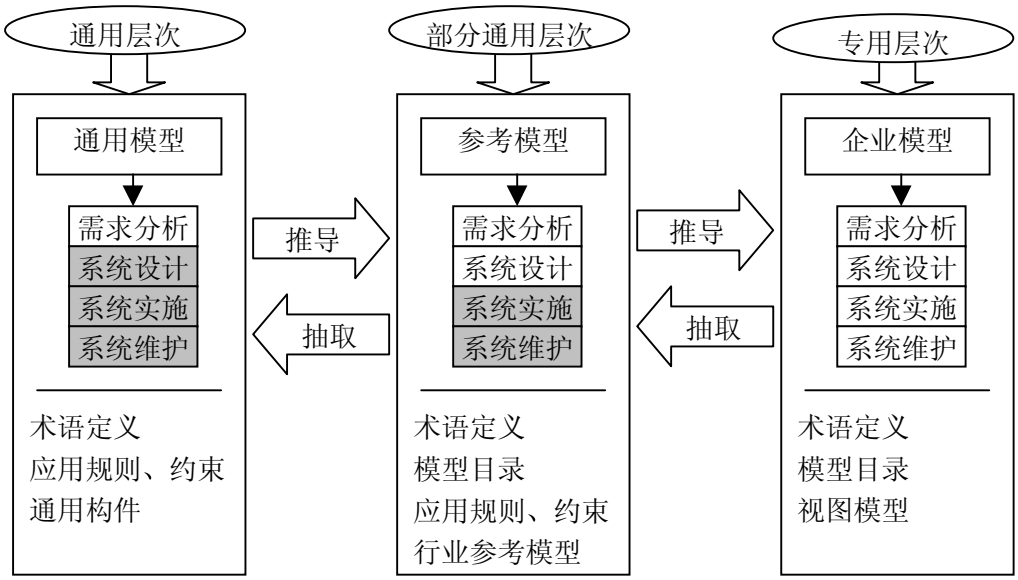


图 11.6 模型从通用到专用的发展过程

本文提出的企业模型在层次上包含一个企业结构模型、几个业务模型和一个数据模型。其中，每个业务模型包含一个工作流模型（过程模型）、一个功能模型、一个信息模型、一个组织模型及一个资源模型。各模型的具体功能及作用如下：

(1) **企业结构模型：**描述了一个多点环境企业的地理位置分布，它可作为模型的整体浏览和创建的基础并创造以后模型应用的多点安装。

(2) **工作流模型（过程模型）：**工作流模型是用来对企业业务过程、设计过程进行可视化

建模，并使用业务功能对相关的业务进行控制。即表示了基本的过程及其控制功能。了完成这些，每个工作流模型中都至少包括几种不同的元素：基本流程（如物资流、信息流、资金流等基本过程）；基本活动（基本流程上的基本活动代表了如原材料的入库、最终产品装配及最终产品的发送等）；业务功能；触发器等。

(3) **功能模型：**描述了企业要实现企业的应用目标所需的功能。采用 IDEF0 方法进行建模。

(4) **信息模型：**描述了企业要实现企业的应用目标所需的信息。采用 IDEF1X 方法进行建模。

(5) **组织模型：**描述了企业组织的层次化结构。另外，也定义了组织中不同的角色。

(6) **资源模型：**描述了企业要实现企业的应用目标所需的各种企业资源。

(7) **数据模型：**描述了每个企业模型物理上和逻辑上的数据模型信息。

另外，在业务模型和数据模型间的一层：对象模型管理是基于以上的各种模型，对它们本身及它们之间的关联、引用、映射及协调它们之间的关系，并对集成化的建模机制进行管理。基于这种对象模型管理,企业的模型可以得到整体的优化和控制。图 11.7 给出了建模系统的模型层次结构图。

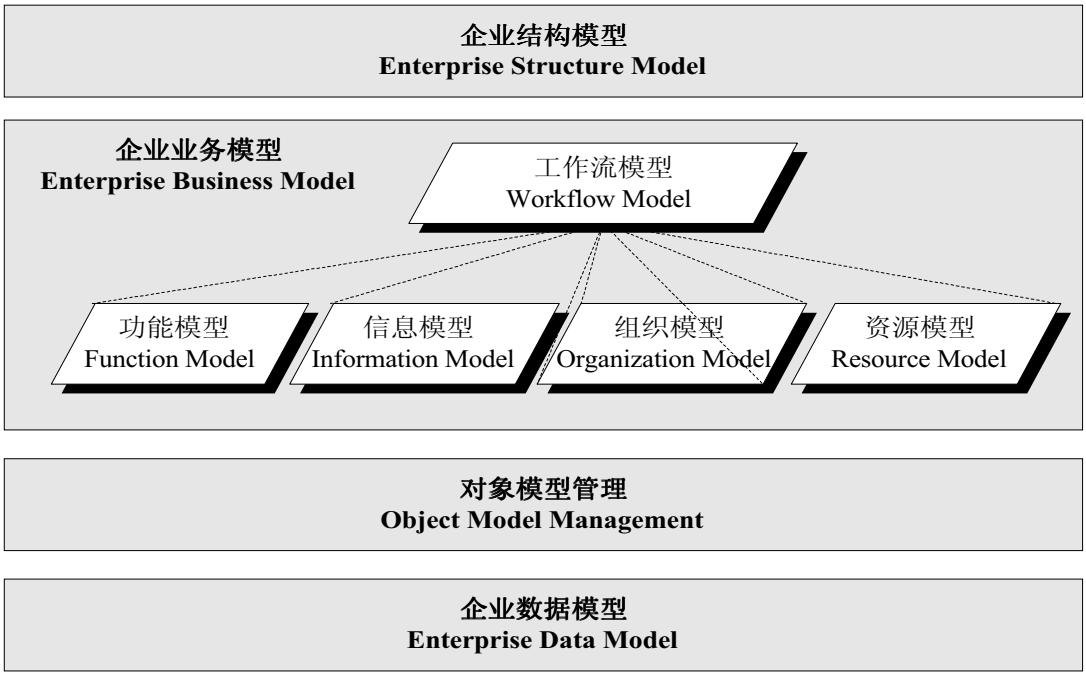


图 11.7 建模系统的层次结构图

11.4.2 基于CORBA的企业建模与优化工具系统

上节介绍了企业建模系统体系结构和建模方法。限于篇幅，有关每个视图模型的建模方

法没有进行深入的介绍。本节在上述体系结构和建模方法的基础上，介绍集成化企业建模与仿真分析系统的开发实施方法。图 11.8 给出了集成化企业建模与仿真分析系统的体系结构图。所给出的集成化建模与分析系统采用基于 CORBA 的 Orbix for WEB 软件总线及其服务为系统支撑环境，应用系统主要由建模工具系统、模型优化工具系统和模型实施工具系统组成。

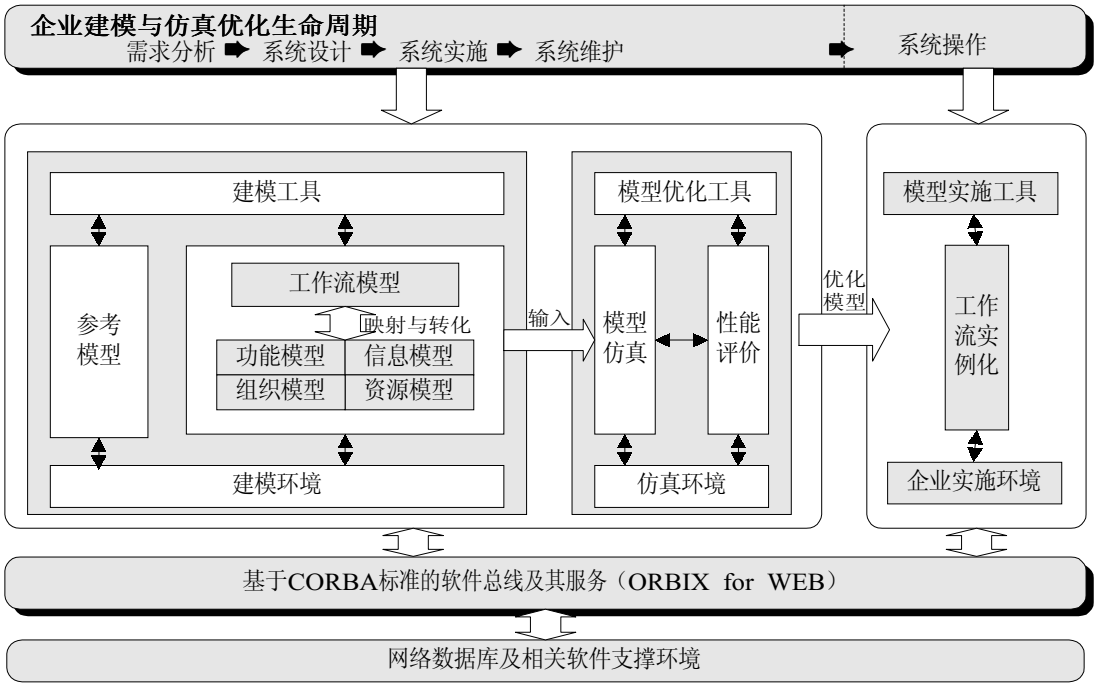


图 11.8 集成化建模与仿真优化分析系统体系结构图

建模工具系统是由建模环境和相应的视图建模工具组成一个集成化系统。视图建模工具系统以过程视图建模工具为核心，功能视图、信息视图、组织视图和资源视图建模工具为辅助，并由模型映射与管理工具实现不同建模阶段与不同视图模型之间的映射与一致性维护。通过建立一个基于 CORBA 软总线的协同建模环境，采用面向对象方法中不同视图对象模型之间的关联，利用对象之间的请求/服务的方式来实现不同视图模型之间的动态连接，从而支持集成化的企业协同建模与仿真优化分析。

在基于 CORBA 标准的 ORB 服务的分布式协同环境支撑下，集成化建模将按照项目研究提出的建模体系结构，以过程模型（包括产品设计开发过程、生产经营过程、产品制造过程的工作流模型）为核心，以其它模型，包括功能模型、信息模型、组织模型、资源模型为辅助模型，建立企业模型。同时，采用面向对象的方法实现不同视图模型之间的集成和导航，即支持企业建模工具系统中不同视图模型之间的映射、不同建模阶段的模型之间的映射与从通用模型到参考模型到专用模型之间的映射。

模型优化工具系统以基于 CORBA 标准的 ORB 服务为支撑，包括集成化的仿真工具、仿真环境和企业性能评价体系。集成化仿真将仍以工作流模型为核心，以 Petri 网技术为后台支撑系统，以基于 CORBA 标准的 ORB 服务为底层支撑技术的仿真策略，通过 Petri 网模型的运行来模拟工作流模型的运转，进一步实现对企业模型的仿真，为系统的分析与评价提供可靠的信息。在仿真的基础上，依据研究得到的企业性能评价体系，建立描述企业运行的良好的性能指标，如：时间（周期）、成本、设备（资源）利用率、瓶颈资源、优化方案/计划、设备空闲比、稳态性能等对仿真结果进行分析，发现存在的问题，以 BPR、产品设计开发过程优化、制造过程优化和资源优化配置的思想进一步指导企业的改进和优化。

模型实施工具系统通过对工作流（过程）模型的实例化和执行服务实现对于企业 CIMS 实施的支持，尤其是实现对于部分需要进行过程集成的运行系统的支持，也可以通过实例化运行分析为企业的 CIMS 实施的不同方案提供比较结果和比较接近企业运行的模拟运行环境，为实施 CIMS 的技术人员提供直观的指导。

本节所介绍的集成化的企业建模方法和工具系统是范玉顺教授负责的 863/CIMS 主题重大关键技术攻关项目“面向 CIMS 工程实施的企业建模和分析系统”的主要研究内容，项目的目标是开发出本节介绍的集成化企业建模工具系统并在 2 个企业进行应用，项目将于 2000 年 12 月完成。

11.5 工作流在CIMS中的其他应用领域

本书的前 9 章对工作流管理技术进行了系统的介绍，第 10 章对工作流管理技术在支持企业经营过程重组中的作用与应用过程进行了详细的介绍，11.3 和 11.4 两节对工作流管理技术在企业应用集成和企业建模中的应用进行了介绍。从本书前面的介绍，读者可以知道工作流管理技术的出现和迅速发展是市场竞争环境对企业组织结构变化与先进制造战略的客观需求，是计算机与网络技术发展等诸多因素作用下的必然产物。它的出现促进了企业的计算机应用水平上升到了一个新的阶段，即从支持企业功能实现的事务处理系统发展到支持企业经营目标的业务处理系统，所以也有人将工作流管理系统称为是企业的业务操作系统（BOS），另外一方面，企业的工作流管理技术的应用也促进了工作流技术的迅速发展。先进的计算机与网络技术的出现与在工作流管理系统实现方法中的应用则大大提高了工作流管理系统的能力和水平，尤其是分布对象计算技术（如 CORBA、DCOM）、分布数据库技术、WEB 技术、代理技术等。

工作流管理技术除了在企业应用集成和企业建模中有良好的应用外，它在企业还有其他很多应用背景。从生产经营的角度来看，企业可以被视为由多个相互关联的不同层次的过程所组成的网络，这些过程相互之间存在着顺序 / 并发、资源共享 / 冲突以及目标相关等关系。在 CIMS 信息环境下，这些过程可以被视为多个相关的工作流。同时，工作流是企业中各种流的载体，它带动了信息流、物料流、资金流的流动，并决定了它们的流速和流量。通过工作流可以方便地考察信息、物料、资金等随过程的变化情况，从而对某些关键指标进行跟踪和计算。

工作流的描述方法不仅清晰、自然，容易被理解和接受，具有很强的描述能力。它综合了企业的多个视图，不仅描述了“做什么”、“怎样做”，而且还定义了“由谁做”、“用什么做”，从而比较全面的反映了企业的生产经营过程。

工作流技术在 CIMS 中有广阔的应用前景，尤其是在当前敏捷制造、并行工程、企业经营过程重组得到企业广泛的认同和重视的情况下，根据我们对工作流技术和 CIMS 应用的了解，工作流技术可以在以下的一些应用领域得到应用并发挥重要作用。

1) **并行工程**：在并行工程中，产品设计过程是工作流技术可以很好的用来描述产品开发过程的建模和管理，它同样可以用来作为产品协同设计、产品设计中的冲突协调、产品数据管理与流程控制的支撑系统。在应用于这个领域时，需要增强对产品数据及其相关的集成文档的描述能力，还需要在工作流技术中融入 CSCW 的技术和方法；
2) **敏捷制造**：工作流管理可以作为企业间信息集成的使能工具，基于 WEB 的和基于邮件方式的工作流管理系统可以为企业灵活的实现动态联盟和信息交换发挥重要的作用。在这个领域中的应用要充分考虑在广域网环境下系统之间消息传递的可靠性问题和不同工作流系统和产品之间的互操作和重构问题；
3) **供应链管理**：工作流管理技术可以较好的用来进行供应链建模和管理功能，结合工作流仿真和优化技术，它还可以用来进行企业分销体系和供应体系的优化。为了能够真正实现优化，还需要加强工作流模型的仿真与优化能力；
4) **企业经营过程重组**：这是工作流技术应用的主要领域。虽然工作流管理为系统的重构提供了必要的手段，但是要真正实现企业经营过程的快速重组，企业的应用系统需要按照组件的方式进行构建或改造，而且对应用组件的粒度要求应该与过程重组要求的灵活性相匹配。即灵活性要求越高，应用组件的粒度应该越小；
5) **企业建模与系统集成**：以工作流模型为核心，功能、信息、组织与资源视图为辅助研究集成化的企业建模方法，并开发相应的集成化企业建模工具，目前这方面的研究工作

已经得到重视，在进行这方面的工作时要解决不同视图模型之间的集成问题和模型的一致性问题。在此基础上进一步可以建立以工作流管理系统为基础的集成平台和集成框架软件，实现方便、快捷、灵活的应用系统集成。

6) **办公自动化、项目管理、成本控制、车间管理：**工作流管理技术可以方便的用于办公自动化、项目管理、成本控制等领域，并可以方便的建立以工作流管理为核心的车间管理与控制系统。

工作流技术综合了计算机科学和管理科学中多个研究领域的原理、方法和技术：数据库管理、C/S 技术、编程语言、图形化用户界面、系统集成、消息传递、文档管理、仿真等等。最近几年企业对于过程建模和 BPR 工具、敏捷制造、并行工程的需求为工作流提供了一个广阔的市场，使得工作流产品得以迅速发展。而且，工作流产品的供应商不断将信息技术、web 等研究中的最新成果应用于自己的产品开发中，使得其能够迅速普及。尽管如此，目前的工作流产品还存在很多问题有待解决。随着工作流技术的进一步发展，它必将在提高企业效率和竞争力，更好地适应市场变化等方面起到举足轻重的作用。