分类号 密级

U D C 编号

中南大學

CENTRAL SOUTH UNIVERSITY

硕士学位论文

论 文 题 目 工作流模型仿真相关技术的研究

学科、专业 计算机系统结构

研究生姓名 夏 媛

导师姓名及专业技术职务 李建华 教授

2007 年 5 月

摘 要

工作流技术已经逐渐成为实现企业业务过程建模、业务过程仿真分析、业务过程优化，最终实现业务过程自动化的核心技术。作为可以与工作流管理系统相集成的外部系统，工作流模型仿真系统在过程定义中的评估、优化作用已越来越引起业界的重视。

本文系统地介绍了工作流模型仿真技术的相关概念、研究的目的和意义及国内外的发展现状，详细阐述了工作流建模和仿真工具的需求分析、设计与实现。分析了目前工作流模型仿真的不足，并从建模、配置和运行阶段构建了一个工作流模型仿真系统（WMSS）。通过扩展工作流管理联盟的传统工作流理论建立一个支持仿真的工作流过程定义元模型，在前人工作的基础上提出了结合工作流历史数据分析的支持仿真的工作流模型，该模型由基础模型和辅助模型组成，对各个子模型给出了形式化的定义。在过程定义元模型的基础上，针对工作流过程定义语言 XPDL 对仿真执行过程不确定性描述的不足设计了支持仿真的工作流过程定义语言 SIM-XPDL，遵循了对 XPDL 的兼容性等原则，提供了对活动的动态/静态执行时间、转移分支概率描述的支持。针对工作流仿真环境的设置，利用工作流历史数据的统计分析结果作为工作流仿真环境参数的参考设置，详细阐述了确定仿真环境参数的步骤，并给出了分析实例。最后设计了工作流模型仿真系统原型，提出构建一个较为通用的仿真引擎，该引擎通过调用工作流引擎提供的接口模拟用户执行任务，并给出了其关键组件的设计，对仿真思想和关键算法进行了描述，通过仿真实例验证了仿真算法的可行性。

本文对支持仿真的工作流模型、过程定义描述语言、工作流仿真环境参数设置及工作流模型的仿真执行等方面进行了研究，最后指出了进一步研究需要解决的问题。

关键词 工作流模型，过程定义，仿真，SIM-XPDL

ABSTRACT

Workflow has become the core technology of enterprise business process modeling, simulation analyzing, optimizing for process automatization. As the external system that could be integrated with workflow management system, workflow model simulation system has become more important in process definition estimation and optimization.

The relevant conceptions, research objectives and meanings, development status in domestic and overseas of workflow model simulation are introduced in this thesis, and the requirement analysis, design and implementation of workflow modeling and simulation tool are expatiated。Based on the deficiency of current workflow model simulation, a workflow model simulation system (WMSS) is designed on basis workflow life cycle which is divided into build stage, configuration stage and execution stage. By extending Traditional workflow theory, one process definition meta model is designed which could support simulation, and a workflow simulation model in combination with workflow history data analysis model is established on basis of the former work, workflow simulation model consists of basic model and supporting model, then the submodel formal expressions are also given. Due to the deficiency of the process definition language XPDL (XML-based process definition language) for describing indeterminacy of simulation execution process, simulation-supported workflow definition language (SIM-XPDL) is designed, which sticks up principles such as compatibility to XPDL. SIM-XPDL supports activity dynamic/fixed execution time and transitions split probability. Given the demands on simulation environment of workflow simulation, an approach for environment simulation parameters reference setup is presented, which makes use of workflow history data analysis. The step of set the simulation parameters is exhausted in detail and the example is given at last. Lastly, workflow model simulation system prototype and universal simulation engine are designed, and simulation engine could call the interface of workflow engine and simulates the user completes workitem. The design of core components and simulation idea and algorithms are

given, and the feasibility of these algorithms is proved by simulation examples.

In a word, simulation-supported workflow model, process definition description language, workflow simulation environment parameters setup and workflow model simulation execution are studied in this paper. Finally their futures are prospected.

KEY WORDS workflow model, process definition, simulation, SIM-XPDL

目 录

第一章 绪论

1.1 引言

1.1.1 问题的提出

目前企业经营过程重组（Business Process Reengineering，BPR）已成为企业深化内部改革，提高竞争力的一个重要手段，但是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，所以人们在真正的重组实践中往往达不到预期的效果，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[1]。这主要是由于有关 BPR 的理论还不够成熟，而且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，无法对重组后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，而工作流技术正好为减少这个重组的风险提供了一个非常合适的解决方案[2][3][4]。

随着现代网络技术的不断发展、企业经营过程重组概念和方法在企业中被重视程度的提高、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改变，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重视。其中作为可以和工作流管理系统相集成的模块，工作流模型仿真对过程定义的评价、分析、优化作用已越来越受到业界的重视。工作流模型仿真是工作流模型具体执行前的模拟过程，利用离散事件驱动的仿真引擎模拟工作流模型中的各项活动，自动推进工作流实例的运行[2]。工作流模型仿真的主要目的就是对定义的过程进行分析和评价，通过动态运行、结合评价指标体系，分析仿真数据，从而发现问题、改进或优化业务过程。在工作流建模技术应需求发展之后，使更复杂的业务过程建模成为可能，从而对工作流模型仿真的需求也更为迫切。

1.1.2 本文研究的背景

工作流技术的概念起源于生产组织和办公自动化领域，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。工作流建模是将现实世界的业务过程抽象出来，并用一种

形式化的、计算机可处理的方式来表示[5]。工作流管理系统（Workflow Management System，MfMS）则是通过将工作活动分解为任务、角色、规则和过程来进行执行和监控，从而达到提高生产组织水平和工作效率的目的。工作流管理系统实施的三个阶段包括过程建模，过程实例化，过程运行。其中，工作流管理系统主要由两个功能组件所组成：工作流建模组件和工作流执行组件[6]。工作流建模组件主要为工作流建模人员提供一个建立时（build-time）环境，使得它们可以定义、分析和管理工作流模型；工作流执行组件的主要功能是为工作流的创建、执行和管理提供一个运行时（run-time）环境。

在企业应用中，工作流经常与经营过程重组相联系，完成对一个组织（或机构）中核心经营过程的建模、评价分析和操作的实施。工作流管理技术经过二十年左右的发展已经日益成熟，BPR、并行工程等已经把工作流作为一种重要的使能技术来对待。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，其相应的工作流过程模型也需要不断改进。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。在其实施的过程中通过对工作流模型进行仿真，找出存在的问题，对模型进行改进得到优化的业务过程模型。

文献[7]中将经营过程中的流程分析和建模工具分为三种：流图工具、CASE工具、仿真建模工具。起初仿真并不具备优势，由于仿真软件与其它建模工具难以集成，并且专用的仿真语言对建模者的技能要求比较高，导致它没有成为业务过程建模的主流。后来的仿真软件考虑到集成和易用性问题才逐渐加以推广，大量的仿真工具也随之出现，其中既有简单易学的流图型工具、也有动力学工具，主要还是离散事件型工具。通过定量分析企业经营过程运行的各项性能指标来判断经营过程是否存在瓶颈或死锁因素，经过过程性能如何。仿真结果可以作为经营过程的评价，过程改进方案的可行性和有效性验证的依据，也是企业决策的可靠基础。但作为工作流技术一个不可或缺的分支，工作流仿真技术的研究迄今为止在全球范围内还非常薄弱。这主要是由于缺乏仿真方法和工具支持、性能指标难以确定、企业行为及业务本身具有很多不确定因素和复杂性。

1.1.3 研究的目的和意义

通过工作流建模工具，过程设计人员可以定义业务过程执行的各个方面。而且工作流建模工具已经发展得比较成熟，过程设计人员可以很方便地建立业务过程的工作流模型。但是如果要设计一个“好”的工作流过程定义依然比较困难，这是由于：

1. 对于工作流过程定义往往需要依靠专业的模型设计人员，而对于用户的需求由于缺乏沟通和理解会导致过程定义不能满足实际的业务需求。

2. 过程定义包含了很多方面的信息（活动、资源、数据等），然而在实际中要对各个方面做到精确定义比较困难。

3. 过程定义是否正确、是否存在瓶颈往往要等实际执行之后才能清楚，对于复杂的过程定义如果要等到实际执行之后再出现问题对整个系统都会造成影响。

4. 过程定义过程通常是独立进行的，但在实际的应用中，在运行时不同的过程实例会相互影响，例如它们会共享资源，调用同一外部应用等，在设计中需要考虑这些影响是很困难的。

从上述分析可见，过程设计人员急需一个工作流仿真工具来帮助他们对过程定义的各项性能进行测试。工作流管理系统的运行通常是先使用建模工具对系统建模，并将模型转化为工作流的过程定义，而后由工作流引擎解释过程定义并生成过程实例来管理工作流的执行。这种机制往往只侧重于工作流模型的定义和执行，管理系统只能在运行一段时间之后才能对模型进行评估，找出模型的缺陷对模型进行修改。仿真技术为这一问题的解决提供了一个行之有效的方法，它可以在系统运行之前对模型进行分析，及早发现工作流模型瓶颈，方便地做出修改设计等决策。因此，工作流仿真技术在这些因素的推动下成为迫切的需求。

1.1.4 基本概念

本文中引用的工作流领域常用名词的相关解释如下：

工作流管理系统（Workflow Management System）：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。

工作流模型（Workflow Model）：以过程模型为核心、包括组织模型、资源模型，描述了一个能够由工作流执行服务软件系统执行所需要的所有信息。

过程模型（Workflow Process Definition）：构成了工作流模型的主体部分，包含了组成工作流模型的所有活动、转移、参与者及相关数据等，其中过程模型通过参与者的角色与组织模型和资源模型相关联。在工作流建模阶段利用一个或多个建模方法及相应的建模工具，完成实际的经营过程到计算机可处理的形式化定义的转换，所得到的定义称为过程模型或过程定义。

过程实例（Process Instance）：是过程定义的一个实例，它表示业务过程的一次执行，对于一个过程定义可以生成多个过程实例。

活动实例（Activity Instance）：过程定义中活动的一个实例，一个活动实例产生一个或多个工作项。

工作流引擎（Workflow Engine）：提供工作流过程定义执行的运行环境。

工作项（Workitem）：可被工作流参与者执行的活动实例的表示，一个活动实例通常产生一个或多个工作项。

工作列表（Worklist）：工作流参与者所拥有的工作项的列表。

1.2 国内外研究现状

近几年来，国内外对工作流仿真技术的研究侧重于两个方向[8]：

第一是工作流仿真性能研究，这个研究方向主要是提出性能指标及相关的分析方法。常用的指标有：完成时间，活动成本和资源利用率等，分析方法有：时间/成本关键路径 TCPM/CCPM（Time Critical Path Method/Cost Critical Path Method）、计划评审技术 PERT（Program Evaluation and Review Technique）、活动成本分析法 ABC（Activity Based Costing）等。

第二是对工作流仿真机制及体系结构的研究，这个研究方向探讨实现仿真的技术及如何建立工作流仿真系统。

目前国内外已出现了多种工作流仿真工具，在这些工具中，有的是将仿真功能嵌入了建模工具中，有的是将仿真功能和执行引擎结合起来，有的开发了专门的仿真系统。日本京都大学开发的基于多代理结构的 Work Web System 工作流管理系统[9]将建模和仿真集成在一起。其缺点是支持的模型比较简单，通用性不好。Micrografx 公司的 iGrafx 建模工具[2]也是将仿真功能嵌入到建模工具中，它通过给图形赋予属性信息和脚本注释，进行工作流过程的模拟。通过 iGrafx 工具，过程定义人员可以很方便地建立工作流模型，并对其进行仿真分析。汉城大学工业工程系开发的 SNUFlow 工作流管理系统[10]包括四个模块：过程定义设计，工作流引擎，工作流客户端和工作流监视器。为了增加仿真功能，他们开发了一个新的模块 WFSIM（workflow simulation）。该模块可以完成从 SNUFlow 过程定义语言到 SIMAN 仿真语言的转换工作，这样可以直接利用 SIMAN 这个成熟的商业仿真工具进行后续的工作。英国曼彻斯特大学计算机系利用角色活动图（Role Activity Diagrams）建立工作流过程定义，并将 RAD 完成转换到 WITNESS 系统的模型映像[11]。吉林大学计算机系提出了基于功能网的工作流模型建模和仿真工具[12]，其功能网是在标准 Petri 网基础上，综合了时延，着色和扩充特性，还有冲突解决与其他一些功能特性。清华大学 CIMS 中心提出了基于工作流的企业过程的建模、仿真、使能系统的模型和体系结构[13][14][15]，它将工作流模型转化为集控 Petri 网实现企业过程仿真。CIMFlow Simulator 是 CIMFlow 软件中的一个工作流仿真工具，它与工作流建模工具 CIMFlow Modeler 一起组成了一个完整的工作流建模及仿真系统。上海交通大学计算机集成技术开放实验室提出了

P_PROCE（Process，Product，Resource，Organization，Control&Evaluation）参考模型[17]，是一个集成化的多视图模型，并且将基于 ECA 规则驱动的仿真引擎和执行引擎结合起来，从执行逻辑上保证系统的可靠性。浙江大学 JTangFlow[8]使用软件测试理论支持工作流仿真过程，提出了基于测试可仿真的工作流模型 JTangFlow。文献[16]中采用层次型的工作流仿真模型，基于离散时间动态系统（DEDS）仿真原理的实现工作流的仿真系统。SIMPROCESS 是由 CACI 开发的过程仿真软件工具[19]，它可以描述业务的动态视图或信息流图，并支持层次化建模与仿真，是一种比较成熟的过程仿真工具。

上述实现工作流模型仿真的方法，有的是利用现有的仿真工具，将工作流模型转换为仿真工具支持的仿真模型从而完成仿真，但是在这个转换过程中会导致信息的丢失。因为尽管工作流模型和离散系统仿真模型较为相似，但它们有各自的特点。在分别开发两种商业软件的时候，并没有过多考虑对方的需求。在这种前提下，模型的转换不得不采取一些假设和一些折中的方法。这样就使工作流仿真的可信度打了一定的折扣，也限制了某些工作流模型的仿真分析。有的是利用自己的建模方式和仿真引擎或是将仿真引擎和执行引擎结合起来，可以按实际需要来仿真业务过程，不必为了迁就已有的软件功能而进行简化和折中，但是系统的通用性非常有限。同时在仿真执行之前通常我们需要对仿真环境、仿真参数进行设置，如果任意设置仿真参数会造成仿真结果的失真。如何进行历史数据的挖掘，从中得到仿真环境的参考设置也是我们需要考虑的问题，而现在的这些工作流工具很少提供了对工作流历史数据的挖掘分析。

1.3 论文的研究概要和内容安排

1.3.1 研究概要

根据工作流模型的仿真执行过程，从建模阶段到仿真执行，整个论文是由建模、验证、仿真环境设置、仿真执行构成的组成的实施框架。

首先对支持仿真的工作流建模的需求进行分析，提出了支持仿真的过程定义元模型。在前人工作的基础上提出了结合工作流历史数据分析的支持仿真的工作流模型，它有机的结合了资源、组织、度量等因素，共分为两个层次，并给出了形式化的描述。

然后通过扩展 XPDL（XML Process Definition Language），研究一种支持仿真的工作流过程定义语言。结合支持仿真的工作流的形式化模型，在遵循兼容性、高内聚性和可扩展性等原则的前提下，通过扩展 WfMC 的 XPDL，定义一种支持仿真的工作流过程定义语言 SIM-XPDL，并对其进行了兼容性分析。

接着将工作流的过程实例历史数据作为观测数据进行统计和分析之后得到仿真环境的参考设置，并给出了分析的实例。上述这种方法适用于已存在的工作流模型，对于新建的工作流模型我们往往需要依靠专家经验等作为仿真环境的参考设置。

最后给出了一个工作流模型仿真原型系统。基于开源项目 Shark、JaWE 等设计了工作流模型仿真系统，对仿真引擎的关键模块进行了详细介绍，描述了相关的仿真算法并给出了仿真实例。

1.3.2 内容安排

论文包括六章：

第一章，绪论：介绍了进行工作流模型仿真研究的背景，分析了国内外研究的现状，指出了研究的意义和目的，并在此基础上提出了本文的研究内容。

第二章，支持仿真的工作流模型：研究了工作流管理系统参考模型，对目前的工作流模型进行了分析，在前人工作的基础上将仿真模型与工作流模型合二为一，提出了支持仿真的工作流模型的基础模型和辅助模型及其形式化定义。

第三章，支持仿真的工作流过程定义语言 SIM-XPDL：在分析原有 XPDL 仿真信息的基础上，扩展仿真信息设计了支持仿真的工作流过程定义语言 SIM-XPDL，提供了活动的执行时间、选择转移等参数的定义，使之可以适应更复杂的仿真场景。

第四章，基于历史数据分析的工作流仿真环境研究：分析了工作流仿真环境，通过对工作流历史数据的分析提供了工作流仿真环境输入参数的参考设置，最后利用实例数据进行了分析。

第五章，工作流模型仿真系统：设计了工作流模型仿真系统。在开源工作流 Shark 的基础上给出了仿真引擎的设计。介绍了主要的关键组件、仿真的具体执行步骤和算法等，最后给出了仿真执行实例。

第六章，总结和展望：总结本文工作，并指出进一步的研究方向。

第二章 支持仿真的工作流模型

为实现工作流模型的仿真，需要研究工作流模型描述的业务过程执行环境下的各种情况及影响因素。本章对现有工作流模型进行了分析，扩展了传统的工作流过程定义元模型，在前人工作的基础上定义了支持仿真的工作流模型的基础模型和辅助模型。

2.1 支持仿真的工作流建模分析

企业经营过程的过程建模是经营过程分析和经营过程重组的重要基础，工作流模型是企业过程的形式化描述，可以被计算机所理解。为了能够对企业业务过程进行仿真，首先需要对企业业务过程进行工作流建模。一般地，工作流模型是由一系列活动（Activity）按照一定的约束关系组成，活动在具体的执行过程中需要使用到一定资源、人员等工作流数据。为此，工作流模型首先需要能够表达企业流程中的一系列活动，其次要能够把这些活动按照一定的约束关系联系起来形成一条完整的过程（Process）。在活动和过程的定义中，需要涉及企业的其他资源，如活动和过程的起止时间，活动需要使用的资源，参与活动的工作人员等。为此，工作流模型除了基础的过程模型之外，还需要包括企业的组织模型、资源模型等。对业务过程建模之后，就能通过工作流引擎将任务信息在发送到合适的资源或人员，实现过程实例的有效运转。

针对不同企业应用、基于现实的不同看法、侧重于某个方面，很多学者或公司都提出过自己的工作流模型。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，如：流程图、状态图、活动网络图等。FlowMark[18]是 IBM 公司开发的一个工作流产品，它是基于活动网络的工作流模型，该模型由一个无自环的有向图构成，包括活动、控制连接、转移条件、数据容器、退出条件、同步条件和任务等语法元素。文献[21]提出了一个基于状态变换的工作流模型，将层次性、并发性和交互性扩展到状态转换图中，使得其支持工作流中的并发任务处理。文献[22]又提出一个触发模型，利用关系捕捉将任务定义成用户可发生的事件集合，工作流就是被包含彼此可触发或可被外部事件触发的任务集合的一个系统。WIDE[20]（Workflow on Intelligent and Distributed database Environment）是欧洲委员会（European Commission）的一个工作流项目，为了使工作流模型在描述信息、组织和资源上的能力更强，WIDE 提出由组织模型、信息模型与过程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复杂的人员。虽然这些模

型千差万别，它们却有许多相似之处，大致可以分为基于活动型（Activity-based）、状态转移型（State-transition）、关系捕捉型（Relation-capturing）和基于通信型（Communication-based）四类[23][24]。对工作流仿真来说，虽然工作流模型包含了业务过程执行的许多信息，但工作流模型仅能提供工作流仿真的部分数据，工作流模型需作许多变动才能适合工作流仿真的需要。这样不仅增加了开发工作流仿真模型的工作量，也导致了两种模型的不一致性，对工作流仿真的实现造成了很大的困难。另外，由于对工作流模型进行仿真之前必须建立需要的仿真环境，即仿真所需的时间、资源、组织及事件等，为仿真模型的运行提供数据并进行相应的响应。在文献[25]中范玉顺等人就提出了一种面向仿真的工作流模型，结合前人的工作和思想，本文也提出了一种结合仿真信息的工作流模型，将工作流模型和仿真模型合二为一、有效集成，简化工作流仿真的实现，在本文中将支持仿真的工作流模型称为仿真模型。

2.2 相关工作

在工作流模型实例执行过程中，各个活动的进行总是伴随着事件的发生。特定的事件触发特定的活动，而活动的结束又引发新的事件。工作流仿真与工作流执行相类似，只是在工作流模型真正投入使用之前，通过仿真模拟资源、事件触发等来模拟工作流的执行过程。对于工作流执行与仿真之间的关系可以如下图2-1 所示：

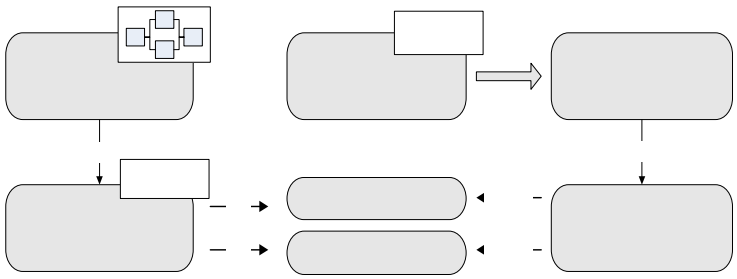

图 2-1 工作流执行与仿真[17]

目前国内外做得比较好的工作流仿真工具包括 CIMFlow simulator 和 SIMPROCESS 等，其中 SIMPROCESS 是目前比较成熟的仿真工具，可以支持工作流联盟过程定义语言 XPDL 模型的仿真，它通过将 XPDL 模型导入 SIMPROCESS 平台下，成为 SIMPROCESS 可以理解的模型，再执行仿真。其中导入成功的条件是工作流模型的访问级别必须为 Public，而且不是所有的 XPDL 模型的所有元素都可以导入 SIMPROCESS [19]。如果想让 XPDL 建模能够为 SIMPROCESS 正确转化，需要扩展逻辑上的节点，包括开始结束节点、AND 和 XOR 型的分支和合并。对于复杂的业务过程来说我们需要在原有工作流模型的

基础上添加专门的活动来充当逻辑节点，使得建模人员的工作量加大。

文献[17]和文献[26]中，根据工作流引擎和仿真引擎的相似性分别以工作流引擎为基础进行仿真功能的扩展构建支持特定应用的工作流模型的仿真引擎。其中仿真引擎的作用是负责解释仿真模型，执行工作流仿真过程。因此在前人工作的基础上，本文希望可以最大限度的重用工作流引擎的执行机制、在尽量不改变工作流模型的结构和功能前提下、根据目前实际的业务需求，对传统工作流管理系统的相关理论进行扩展，建立了支持仿真的工作流过程定义元模型及仿真引擎。仿真引擎通过调用工作流引擎的提供的 API，模拟用户执行工作流实例，自动推进工作流实例的进行。

使用这种方法的优点是：不需要对模型进行转化或改变工作流模型的结构和功能，进一步提高仿真的可信度；可以最大限度的重用工作流引擎的执行机制，不需要改动工作流引擎，在这个基础上进行的仿真使之可以更加符合工作流引擎的性能实际状况。但仍然存在缺陷，因为尽管是对工作流管理联盟提出的过程定义元模型的改进，但目前存在多种工作流模型，在本文中只是针对了工作流管理联盟提出的工作流模型，通用性仍然有限。

2.3 支持仿真的工作流过程定义元模型

根据工作流管理系统实施的阶段，我们对支持仿真的工作流管理系统总体结构从建模到过程执行进行设计。在 WfMC 定义的工作流管理系统参考模型如图 2-2 所示。

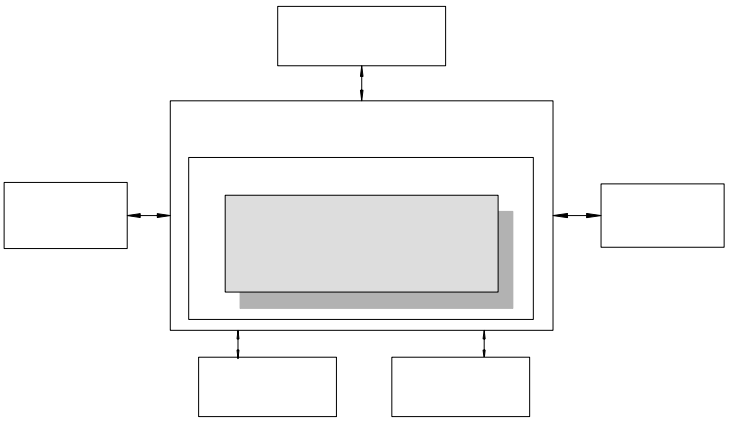

图 2-2 工作流管理系统参考模型[2]

过程定义工具：提供了工作流过程定义的服务，它输出可以被工作流执行服务所识别的过程定义。

工作流执行服务：由一个或多个同构或异构的支持仿真的工作流引擎组成，除了保留了用于创建、管理和执行工作流模型实例的软件服务之外。应用系统可

以通过工作流应用编程接口 API 来访问工作流执行服务。

工作流客户端应用：是工作流的实现组件，通过它用户可以激活与各种工作流模型活动相关的客户端应用系统，实现与工作流执行服务的交互。

工作流调用的应用：被工作流执行服务激活的用于实现工作流活动内容的应用系统。

工作流管理工具：提供工作流管理系统管理和监控的工具，包括用户管理、角色管理、审计管理、资源管理、过程监控管理等。

在该参考模型中还定义了 5 个接口，用于定义组件间的互操作规范[27]~[30]：接口 1 为工作流执行服务和工作流建模工具间的接口；接口 2 为工作流执行服务与客户端应用之间的接口，这是最主要的接口规范，它约定所有客户端应用与工作流执行服务之间的功能操作方式；接口 3 为工作流执行服务和直接调用的应用程序之间的直接接口；接口 4 为支持仿真的工作流管理系统之间的互操作接口；接口 5 为工作流执行服务和工作流管理工具之间的接口。通过这五个接口，工作流管理系统可以同外部的软件工具进行交互，这些接口一般通过 API 的形式提供给用户或软件开发商，因此也称这些 API 为 WAPI（Workflow API）。

针对工作流建模，工作流管理联盟定义了工作流的过程定义元模型，过程定义元模型描述了工作流模型的元素、元素属性及元素之间的关系，对于工作流模型仿真来说，仿真信息已经是一个不可缺少的元素，因此在扩展 WfMC 所定义的工作流过程定义元模型[5]的基础上，本文定义的支持仿真的工作流的过程定义元模型如图 2-3 所示。其中填充色为灰色的部分是为了支持仿真而改进和添加的元素。

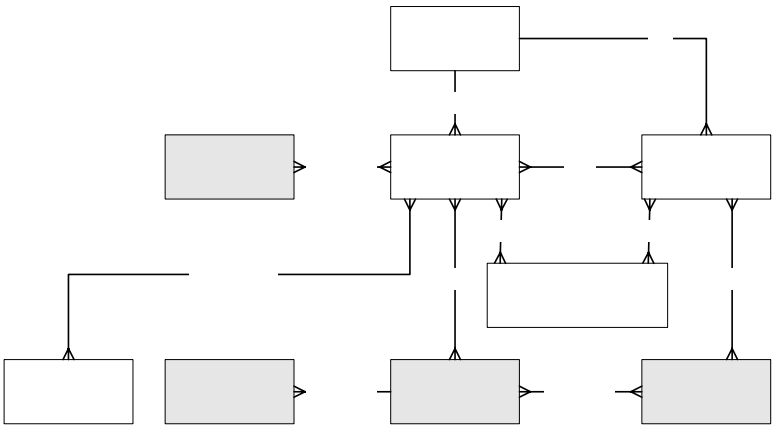

图 2-3 支持仿真的工作流过程定义元模型

扩展后的实体包括工作流过程定义（Workflow Process Definition），活动（Activity）、工作流相关数据（Workflow Relevant Data）、被调用的应用（Invoked Application）、转移（Transition）、转移条件（Condition）、分支选择概率（Split

Probability）、角色（Role）、仿真信息（Simulation Information）。

工作流过程定义（Workflow Process Definition）：描述业务过程的活动（Activity）以及活动之间的控制流和数据流逻辑。

活动（Activity）：完成业务过程的一个逻辑步骤。活动的类型不同，其执行方式也不一样。如果是人工交互型活动，则需要引用角色（Role）概念；如果是自动型活动，则将调用应用程序（Invoked Application）；仿真过程中，对于自动型活动就沿用原有机制进行调用，我们只需要对人工型的活动进行自动执行，调用仿真信息（Simulation Information）来设定活动执行的时间。在执行过程中可能要使用或更新工作流相关数据（Relevant Data）。在一个活动执行成功后，将根据其后续的转移（Transition）的条件（Condition）来决定做哪个活动；在仿真过程中，对于决策型输出根据给出分支选择执行的概率（Split Probability）来判断选择执行哪条分支。

工作流相关数据（Workflow Relevant Data）：工作流管理系统根据工作流相关数据来确定过程实例状态转换的条件。

被调用的应用（Invoked Application）：描述了用于完成业务过程所采用的工具和手段。

转移（Transition）：描述活动（Activity）间的关系，可能包含有转移的条件（Condition）或者分支概率（Split Probability）。

条件（Condition）：定义从当前活动到下一个活动流转或状态转移的规则。

分支概率（Split Probability）：定义了选择执行分支转移的概率。

仿真信息（Simulation Information）：定义了仿真时活动的执行时间。

2.4 支持仿真的工作流模型的形式化定义

2.4.1 支持仿真的工作流模型

在文献[8]和文献[25]中，范玉顺等人通过分析工作流模型和仿真环境的关系，提出了一种面向仿真的工作流模型，该模型结构分为基础模型和辅助模型两个层次，其中辅助模型构成了仿真环境。本文在此基础上，对其中的辅助模型给予相应的扩展，其中基础模型就是过程模型，辅助模型除了资源模型、组织模型之外，还扩展了时间表模型、事务模型、历史数据模型、评价模型。其中时间表模型、资源模型、组织模型、事务模型构成了工作流模型仿真的仿真环境，历史数据模型和评价模型用于仿真环境的参考设置和仿真结果的分析。

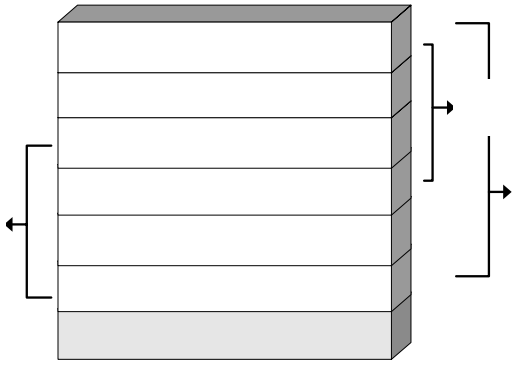

图 2-4 支持仿真的工作流模型

2.4.2 基础模型

基础模型是仿真执行的基础，原有工作流模型的定义中包括了过程模型，资源模型和组织模型，在本文中将过程模型作为基础模型，过程模型通过过程、活动、转移来描述业务过程的逻辑依赖关系。

定义 2.1：过程 Process，P=<id,A,T,RD,PA,AP,Attr>是一个七元组。其中 id 是过程 id 为过程的唯一标识。A＝{a1,a2,…,an}，表示过程定义中所有活动的集合。T 是 Process 中所有活动间转移（Transition）的集合。RD 是工作流相关数据（Relevant Data）的集合。PA 是参与者（Participant）集合。AP 是过程需要调用的应用（Application）的集合。Attr 是过程属性的集合，它是一个六元组<name,accl,version,author,created,EA>，其中 name 为过程名，accl 为过程的访问级别，枚举项为 public 或 private，如果是 public 过程能被外部系统或应用程序调用，如果为 private 则为私有过程；author 是过程的创建者；version 是过程的版本号；EA 为过程的扩展属性集，EA={ea1,ea2,…,ean}。

定义 2.2：活动 Activity，A=<id,type,Implementation,performer,SE，transtrict,SimulationInformation,Time,Attr>为一个九元组。其中 id 为活动 id，type 为活动的类型，枚举项包括 root、subflow、manual 和 automatic，分别代表虚活动、子过程活动、人工交互活动和自动化活动，Implementation 是是活动的实现方式，如果 $type = manual \vee type = root$，Implementation 取值为空；如果 $type = automatic \vee type = subflow$，Implementation 为二元组<id,ActP>，id 是引用的 ap，subflow 的 id，ActP 是实参集合。performer 是活动的执行者，$\exists pa_i (pa_i \in PA \wedge pa_i.id = performer)$。SE 为活动触发和结束的模式，属性包括 manual 和 automatic；transtrict 是活动的转移约束，提供更多的约束和转移描述的上下文相关语义，包括活动的转入转移 Join 和转出转移 Split，转移类型包括 XOR 和 AND。SimulationInformation 描述了仿真时设置的活动执行时间。Time 为活动的时间约束为一个二元组<DL,Limit>，DL 为活动的最终期限（Deadline）

集合，DL={$deadline_1$,$dealine_2$,…,$deadline_n$}。Limit 是活动期望的持续时间。Attr 为活动的基本属性，为一个三元组<name,desc,pri,EA>，其中 name 为活动名称，desc 为活动描述，pri 为活动的优先级别，取值范围为｛0,1,2,…,n｝，数字越大优先级别越高，EA 为活动的扩展属性集合，EA={ea1,ea2,…,ean}。

定义 2.3：转移 Transition，T=<id,name,from,to,type,exp,Attr>为一个七元组。其中，id 为转移 id，name 为转移名称，from 为转移的源活动，to 为转移的目的活动。type 为转移的类型，枚举项包括 normal、exception、otherwise、defaultexception 和 condition。exp 为转移条件的表达式，如果 $type = condition \wedge \exp = true$，转移会被执行。$type = otherwise$ 说明此转移为缺省转移，如果其他转移的转移条件都不满足时，则执行此转移。$type = exception \wedge \exp = true$ 则执行该转移。$type = defaultexception$ 为异常缺省转移，如果产生了异常，并且其他异常转移的转移条件都得不到满足则执行此转移。Attr 为转移的基本信息，是一个三元组<name,desc,EA>。name 为转移的名称，desc 为转移的描述，EA 是转移的扩展属性集合。

定义 2.4：工作流相关数据 Relevant Data，RD=<id,level,type,initial,Attr>为一个五元组。其中 id 为相关数据的 id，level 为相关数据的权限约束，枚举项包括 process 和 activity；type 为相关数据的类型；initial 为 rd 的初始值；Attr 为相关数据的基本属性为<name,desc,length,EA>是一个四元组，其中 name 为相关数据的名称，desc 为数据描述信息，length 为数据长度，EA 为扩展属性集合。

定义 2.5：仿真信息 Simulation Information，Simulation Information＝<id,worktime,value>为一个三元组。其中 id 是该仿真信息所属活动的活动 id，worktime 描述该活动的执行时间类型，枚举项包括 FixedTime 和 DynamicTime，Value 是执行时间的值域。

定义 2.6：应用程序 Application，AP=<id,name,er,fp,EA>为一个五元组。其中 id 为调用的应用的 id，name 为应用名称，er 为外部应用程序签名规范的参考，fp 为应用所需的形式参数集，它是通过调用接口与应用程序交互的参数列表，EA 为应用程序的扩展属性集合。

定义 2.7：参与者 Participant，Part=<id,name,type,EA>为一个四元组。其中 id 为参与者 id，name 为参与者名称，type 为参与者类型，枚举项包括 organization、role、human、system 和 resource。EA 为参与者的扩展属性集合。

2.4.3 辅助模型

辅助模型包括事务模型、时间表模型、组织模型、资源模型、评价模型、历史数据模型六个部分，它为基础模型提供工作流模型仿真所需的仿真环境以及仿

真环境的参考设置及评价标准。所谓仿真环境，是描述处理业务过程所运行的实际企业环境，它主要包括过程运行的企业内部环境设置，包括时间表定义、资源定义、组织定义等；过程运行的外部环境定义，主要是指事件生成器的设置以及过程运行的评价；历史数据模型为工作流仿真模型的仿真参数提供参考设置，该模型与基础模型的仿真信息相关联。下面从六个辅助模型的角度来描述这些仿真环境。

1. 时间表模型

定义了工作流模型仿真执行时各种时间设置及仿真过程中企业人员的活动和设备的使用遵循的时间表，在工作流模型管理和过程实例运行等方面都有重要的作用。在 XPDL 定义中有自带的时间设置属性（包括 deadline 和 limit），此外为了能够模拟企业内部环境我们还需要定义资源的作息表（worktimeTable）。在企业中人员的活动和设备的使用遵循一定的工作时间表，比如某一企业生产活动的工作时间表是正常班，即工作日是周一到周五（除去国家法定休息日），作息时间为上午 8 点至 12 点，下午 1 点至 5 点，而售后服务活动的工作时间表则可能 3 班倒，从而保证随时为客户解决各种问题。只有在时间表定义的范围内，资源才是可用的或活动才可被执行。时间表模型就是为模拟企业的真实环境，对企业中的信息时间进行的一种抽象。当活动调用某一时间表模型时，活动的执行作息时间就必须与时间表模型中的作息描述一样。

定义 2.8：最终期限 deadline，DL=<id,type,exp,exname>为一个四元组。其中 id 为 deadline 所属活动的活动 id。Type 为最终期限的执行类型，枚举项包括异步（asynchronous）和同步（synchronous）。若设为同步，当活动到达期限条件时活动还未结束的话，该活动会被终止（terminate），process 走向下一个活动。若设为异步，当活动到达期限条件时活动还未结束，抛出异步异常，并激活类型为异步异常的转移所指向的活动，但当前活动不会被终止。exp 为用于描述期限条件的表达式。遵循一定的语法，设置活动在某个时间之后如果还未完成就抛出异常。其中时间的参考点可以有 PROCESS_STARTED_TIME ：process 被启动的时间； ACTIVITY_ACTIVATED_TIME ： 活动被激活的时间；ACTIVITY_ACCEPTED_TIME：活动被接收的时间；exname 用于描述抛出异常的名称，枚举项包括异步异常（YNC_EXCEPTION）和同步异常（ASYNC_EXCEPTION），执行的异常中可以包含多个异步异常，比如异常名称可以命名为 SYNC_EXCEPTION1、SYNC_EXCEPTION2 等，但只允许有一个同步异常。

定义 2.9：期限 Limit，Limit=<id,value>是一个二元组。其中 id 为该期限所属 activity 或 process 的 id，value 为描述期限的值，对于 process 的期限其参考时

间就是 process 被激活的时间，activity 的期限就是 activity 被激活的时间。在执行过程中，当超过指定时间时会出现超时的提示信息。

定义 2.10：作息表 WorktimeTable，WorktimeTable＝<id,time,desc>是一个三元组。其中 id 为作息表所属资源的 id；在一个企业中每种资源都会有属于自己的作息表；time 为作息时间表的具体设置信息；desc 为作息时间表的描述信息。

2. 事务模型

工作流模型的执行需要驱动力，事务可以理解为来自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。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，而在工作流仿真过程中，事务是需要由计算机模拟产生的，因此，定义仿真环境时要选择相应的事务模型，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。事务设置的不同会对仿真结果产生重要影响。一般来说，定义事务模型时要确定事务产生的规律、数量。针对不同的仿真目的我们设定了 3 种类型的仿真事务模型，它们分别是：间隔型事务模型、完成型事务模型、需求型事务模型。其中间隔型事务模型的事务产生规则是：一个事件发生后，间隔一定时间下一个开始事件才发生。运用该类型的开始事件可以模拟客户到达等现实情况。系统允许用户定义开始事件的发生频度（即相邻事件的间隔时间）和每一个事件发生时的事务数量。间隔时间可以定义为常数，也可以定义为服从一定时间段上的平均分布或正态分布或泊松分布等。完成型事务模型的事务产生规则为：开始事件发生后，直到该事务被处理完，才产生另一个开始事件，开始下一次仿真运行。运用该类型的开始事件可统计经营过程的周期时间。需求型事务模型的事务产生规则是：只要过程的第一个活动空闲，就会有一个事务到达，选择此种事务发生可以验证过程模型的最大输出量。因此，根据上述分析本文给出的事务的定义如下。

定义 2.11：事务 Transaction，Trans=<id,type,timeset,maxnum>为一个四元组。其中 id 为事务发生器的 id；type 为事务发生类型，枚举项包括间隔型、完成型和需求型三种；timeset 为产生事务的间隔时间设定，枚举类型包括 fixedtime（固定时间）和 dynamictime（动态时间），其中动态时间是根据分布类型来随机事务间隔；maxnum 为设置的最大事务数。

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，本文中将除人以外的所有物质实体都称为资源。资源模型主要定义与工作流仿真执行有关的资源信息。我们引入资源类型和个体资源的概念从企业全局的角度定义资源，包括资源的类型，个体资源的数量、成本（其中包括标准成本，超时成本）、所属组织单元等。为

了使仿真系统具备调度能力，资源定义还包括资源的调度规则，即当多个活动同时请求占用某个资源时，会出现排队现象，资源对于排队中的活动按照一定的规则进行选择。资源定义时的调度规则可选择先到先服务（FIFO）、随机选择（Random）、优先级（PRI）等规则。

定义 2.12：资源 Resource，Resource=<id,type,num,orgnazition,cost,control,desc>为一个七元组。其中 id 为资源 id；type 为该资源所属的资源类型；num 为资源的数量；orgnazition 为该资源所属的组织机构的 id；cost 为资源的使用成本，是一个三元组<fcost,scost,ocost>，fcost 为固定成本、scost 为标准成本和 ocost 为超时成本；control 为资源的分配规则，当出现多个活动同时请求某个资源的情况时，就会出现排队情况，这时资源对于排队中的活动按照一定的调度规则进行选择，枚举项包括 FIFO、Random、PRI；desc 为资源的描述信息。

4. 组织模型

组织模型用来定义企业中人的组织形式的模型，在工作流仿真中组织模型主要定义与工作流模型虚拟执行有关的组织信息。在 WfMC 所定义的规范中没有对组织模型进行详细的描述，只是在参与者（Participant）和角色（role）上加以区分，并在此基础上建立一个具有层次化结构的角色模型（role model），这种模型比较简单，对于组织中人员的属性并没有进行描述。本文首先从企业的组织结构角度描述企业，包括组织单元（organization）的划分、每个组织单元的名称、功能描述以及各个组织单元间的层次关系。其次组织模型中详细描述组织中人员（user）的成本属性（其中包括固定成本，标准成本，超时成本，最大超时）和技能属性。同样，组织模型中的人员也存在一个活动分配规则的问题，当出现某个活动等待某个角色、职位或者具体人员处理时，也会出现排队现象，此时，组织模型中的人员将对排队中的活动按照一定的调度规则进行选择，其调度规则与资源定义中的调度规则类同。在组织模型中，用户是组织模型的基本单元，属于某个角色，并具有相关的属性，比如职位、能力等。用户可以是单个用户也可以属于某个组织。

定义 2.13：人员 User，User=<id,name,role,organization,cost,contol,ability,desc>为一个八元组。其中 id 为人员 id；name 为人员名称；role 为人员的角色，通过角色可以约束人员的系统权限；organization 为人员所属组织的 id；cost 为人员的成本属性，是一个三元组<fcost,scost,ocost>，fcost 为固定成本、scost 为标准成本和 ocost 为超时成本；control 为人员的分配规则，当出现多个活动同时请求某个人员的情况时，就会出现排队情况，这时人员对于排队中的活动按照一定的调度规则进行选择，枚举项包括 FIFO、Random、PRI；ablility 为人员的执行能力，用于描述用户能力的不确定对活动执行的影响，可以划分为积极型、一般型、

懒惰型等，并可以设置某种类型的用户完成任务的百分比时间，如积极型：1/3 执行时间，一般型：2/3 执行时间；懒惰型：直到活动设定的 Deadline 或 Limit ；desc 为人员的描述信息。

定义 2.14：组织 Organization，Organization=<id,name,role,f,User,desc>为一个六元组。其中 id 为组织 id；name 为组织名称；role 为该组织结构的系统权限；f 为组织的父组织或上级组织；User 为该组织所直属的用户列表；desc 为组织的描述信息。

5. 评价模型

工作流模型仿真的目的是找出过程模型是否有不合理的地方，指导对过程模型加以改造和优化。通过对不同过程模型的仿真执行，需要利用评价指标对工作流模型做出不同的评价。评价模型就是辅助决策者对工作流模型仿真执行过程中进行评价管理，描述模型中可能涉及的资源以及成本的产生、消耗、转换。对关键性指标（时间、成本等）进行评价，并根据评价的结果对过程模型进行控制，管理，支持工作流模型的重组改造，最终实现业务过程的优化运行和管理。其中用于仿真分析工作流模型性能的数据包括活动的执行时间、资源的使用时间、活动的成本等。

定义 2.15：评价 Estimation，Estimation=<type,result,standands>为一个三元组。其中 type 为评价的类型，分为过程模型功能（function）、结构（structure）和性能（performance）（包括时间、成本）的评价。result 为评价的结果值，standands 是定义评价模型的标准描述。

6. 历史数据模型

在仿真中，仿真环境的设置是否符合实际的情况是仿真结构可信的前提。工作流历史数据详细地记录了工作流实际的执行情况，包括过程、活动、资源和状态信息等，通过对历史数据的分析和挖掘可以得到我们进行仿真环境的参考设置。其中历史数据模型主要描述了指定参数的数据信息以及分析的结果等，可以作为事务模型、时间表模型和基础模型仿真信息的参考设置。

定义 2.16：历史数据 HsitoryData，HsitoryData=<id,r,v,f>为一个四元组。其中 id 为指定的历史数据标识，可以是一个表达式，用于历史数据的提取；r 为描述指定标识的历史数据的属性集，对于不同的历史数据有不同的属性集，比如提取的过程实例信息就会包括过程启动时间、完成时间等多个属性；v 为通过指定标识提取出的历史数据记录的值域集；f 为描述历史数据分析结果的函数或值。

2.5 小结

本章的主要工作是为工作流模型仿真的研究提供一定的理论基础，通过对支

持仿真的工作流建模的分析，在原有工作流管理系统参考模型的基础上提出了支持仿真的工作流过程定义元模型。重点介绍了支持仿真的工作流模型的结构及其形式化的定义，此模型结构分别由基础模型和辅助模型组成，辅助模型中的事务模型、时间表模型、资源模型及组织模型组成了仿真所需的仿真环境，历史数据模型为仿真提供了仿真参数的参考设置，评价模型为仿真结果提供了分析及评价的标准。

第三章 支持仿真的工作流过程定义语言 SIM-XPDL

现有支持仿真的工作流过程定义语言非常少，WfMC 的 XPDL 中提供了基础的仿真信息（Simulation Information），作为一个标准、通用的工作流定义语言 XPDL 仍然不能完全满足仿真和复杂业务过程的需求。由于实际的业务过程中会出现很多不确定性因素，本章通过扩展和修改 XPDL，定义了支持仿真的工作流过程定义语言 SIM-XPDL，在 SIM-XPDL 中扩展了原有的仿真信息使之能够满足我们目前业务过程仿真的需求。

3.1 现有工作流过程定义语言分析

目前常见的工作流过程定义语言有 PSL（Process Specification Language）、WFSL（Workflow Specification Language）、WPDL（Workflow Process Description Language）、XPDL、BPML（Business Process Modeling Language）、BPEL4WS（Business Process Execution Language for Web Services）等，其中基于 XML 的工作流过程描述语言是过程定义语言中的主流[31]~[34]。

PSL 是美国 NIST（National Institute of Standards and Technology）于 1995 年为了解决异构系统之间过程信息集成问题实施的过程规范语言，其中也包括工作流过程定义[31]。PSL 是基于本体论及 KIF（Knowledge Interchange Format）语法的标准语言格式，由于许多产品都开始或已经支持了 XML，为了缩小数据共享的瓶颈，NIST 给出了 PSL 映射为 XML 的准则。但是由于 KIF 的原因，无法将 PSL 的过程描述完整地转化为 XML 形式[32]。

WFSL 是在 Georgia 大学的 METEOR 项目中提出的一种过程定义语言[33]，它是一种基于规则陈述的语言，用来描述多个任务间的应用级的交互。与它配合的是任务定义语言 TSL（Task Specification Language）[35]，TSL 描述任务间相互的依赖关系、数据格式、数据交换、错误处理和数据恢复。这两种语言合并为 WIL（Workflow Intermediate Language），他们所提出的语言中对事务、模型的动态建模和任务间的数据交换做出了许多有益的研究工作。

BPML 是国际标准化组织“业务过程管理促进会”BPMI（Business Process Management Initiative，BPMI）制定的基于 XML 的过程定义语言[36]，定义了一个表达业务过程及实体的抽象模型和 XML 语法。BPML 本身未定义任何应用语义，只是定义了一个抽象模型和表达业务过程的语法，这使得 BPML 有多种用途，包括定义企业业务过程、定义复杂的 Web 服务以及合作伙伴的协作等。

BPEL4WS 是 Microsoft，BEA，IBM，SAP&Siebel 四家公司联合发布的有

关 Web 服务的规范语言[37]，它是早期微软 XLANG（基于块结构的过程语言）和 IBM 的 WSFL（基于图形的过程语言）的综合版本。BPEL4WS 不是严格的过程描述语言，主要侧重于描述 Web 服务模型。

在工作流管理联盟中所提出的工作流管理系统参考模型中定义了五类接口，有关过程定义的引入和导出构成了接口 1 的主要功能。接口 1 定义一个公共的交换格式，使得不同的工作流定义可以实现模型交换，它还实现了工作流的定义和执行两个不同阶段的分离，使得工作流的定义工具和执行工具相互独立，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引擎解释执行。WPDL 是 WfMC 发布的关于参考模型接口 1 的早期标准，也是第一个过程定义语言[27]。

工作流管理联盟在 WPDL 基础上经过发展，并采用 XML 语言描述格式，形成了现在的 XPDL。2005 年 WfMC 发布了 XPDL 的最新版本 XPDL 2.0[38]，比之前的 XPDL1.0 在语法方面增强了很多。

虽然工作流建模方面已经取得了很多的成果，但是对于模型仿真方面还比较薄弱，支持仿真的工作流过程定义语言可以说少之又少，在 XPDL 中提供了“simulation Information”元素的定义[5]，但是目前基本上还没有工作流引擎实现了对这种过程定义语言内建仿真机制的支持[39]，并且 XPDL 中提供的仿真信息也不能完全满足实际仿真的需求。XML 是一种允许用户创建自己的标记语言的元标记语言，具有良好的可扩展性，并且在 XPDL 中已提供了“Simulation Information”的定义，因此本文认为，在遵循原有标准的前提下，通过对传统工作流过程定义语言 XPDL 的深入研究和分析，继承其优势，进行支持仿真的过程定义语言的扩展来满足面向仿真的工作流建模的需求，是比较可行的方法。

3.2 XPDL 简介

目前有很多比较成功的工作流产品，如 Lotus Workflow[40]、CIMFlow-Designer[41]等，它们都有自己内部私有的工作流模型的存储格式，因此难以实现系统之间的工作流过程定义的转换。

作为一个标准、通用的工作流定义语言，XPDL 提供了一般意义下的公共交换格式[42]，其中 Enhydra JaWE[43]、Workflow Model Tool 和 nezha[44]等这些建模工具都是基于 WfMC 的 XPDL 的建模工具。

XPDL 是一种基于有向图的描述语言，其中 Process 由若干 Activity 组成，通过 Transition 将多个 Activity 连接起来。XPDL 包含了很多在过程定义转换中需要的元素，但是在某些情况下，过程定义可能需要一些另外的信息（如多实例情况[45]），而我们要尽可能的用标准的实体或属性集进行过程定义，XPDL 中的扩展属性（ExtendAttribute）就为开发者提供了这样的一个接口，XPDL 中的所

有实体都可以使用扩展属性，开发者通过扩展属性进行需要的功能或其它方面的扩充。

对于仿真方面 XPDL 提供了相关元素的定义，其中 XPDL 中原有的过程仿真信息的 schema 如下：

```
<xsd:element name="SimulationInformation">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element ref="xpdl:Cost"/>
      <xsd:element ref="xpdl:TimeEstimation"/>
       <xsd:any namespace="##other" processContents="lax" minOccurs= "0" maxOccurs="unbounded"/>
    </xsd:sequence>
    <xsd:attribute name="Instantiation">
      <xsd:simpleType>
        <xsd:restriction base="xsd:NMTOKEN">
          <xsd:enumeration value="ONCE"/>
          <xsd:enumeration value="MULTIPLE"/>
        </xsd:restriction>
      </xsd:simpleType>
     </xsd:attribute>
    <xsd:anyAttribute namespace="##other" processContents="lax"/>
  </xsd:complexType>
</xsd:element>
```

时间估算（TimeEstimation）定义如下：

```
<xsd:element name="TimeEstimation">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element ref="xpdl:WaitingTime" minOccurs="0"/>
      <xsd:element ref="xpdl:WorkingTime" minOccurs="0"/>
      <xsd:element ref="xpdl:Duration" minOccurs="0"/>
       <xsd:any namespace="##other" processContents="lax" minOccurs= "0" maxOccurs="unbounded"/>
    </xsd:sequence>
    <xsd:anyAttribute namespace="##other" processContents="lax"/>
```

```
  </xsd:complexType>
</xsd:element>
```

活动执行代价（Cost）定义如下：

```
<xsd:element name="Cost">
  <xsd:complexType>
    <xsd:simpleContent>
      <xsd:extension base="xsd:string">
      <xsd:anyAttribute namespace="##other" processContents="lax"/>
      </xsd:extension>
    </xsd:simpleContent>
  </xsd:complexType>
</xsd:element>
```

等待时间（WaitingTime）定义如下：

```
<xsd:element name="WaitingTime">
   <xsd:complexType>
     <xsd:simpleContent>
       <xsd:extension base="xsd:string">
         <xsd:anyAttribute namespace="##other" processContents="lax"/>
       </xsd:extension>
     </xsd:simpleContent>
   </xsd:complexType>
</xsd:element>
```

工作时间（WorkingTime）定义如下：

```
<xsd:element name="WorkingTime">
  <xsd:complexType>
    <xsd:simpleContent>
      <xsd:extension base="xsd:string">
          <xsd:anyAttribute namespace="##other" processContents="lax"/>
      </xsd:extension>
    </xsd:simpleContent>
  </xsd:complexType>
</xsd:element>
```

预期持续时间（Duration）定义如下：

```
<xsd:element name="Duration">
```

```
    <xsd:complexType>
      <xsd:simpleContent>
        <xsd:extension base="xsd:string">
          <xsd:anyAttribute namespace="##other" processContents="lax"/>
        </xsd:extension>
      </xsd:simpleContent>
    </xsd:complexType>
</xsd:element>
```

其中元素的表示意义如下表 3-1 所示。

表 3-1 Simulation Information 元素的表示意义

元　素	描　述
Cost	执行该活动的平均代价
Instantiation	定义一个活动可以被实例的次数：ONCE 或者 MULTIPLE
Duration	执行一个活动实例的预期持续时间
WaitingTime	平均等待时间
WorkingTime	平均工作时间

3.3 XPDL 在支持仿真方面的不足

现在支持仿真的工作流过程定义语言比较少，往往是把工作流模型转化为其他形式（如 petri 网）表示的模型之后运用仿真工具实现仿真，或是采用将工作流模型转化为仿真语言（如 SIMAN 等）。如果单独地提出一种仿真语言或将 XPDL 转换为其它仿真工具可以接受的形式，其工作量也是比较大的。在遵循现有通用的国际标准的前提下，通过对 XPDL 的深入分析和研究，继承其优势，并适当进行扩展使之能够支持我们仿真需求的过程定义语言，这是一种比较可行的方法。

XPDL 提供了一系列的仿真元素定义，这些信息存在于活动的 Simulation Information 中。虽然 XPDL 提供了仿真元素的定义，但是在实际的业务流程执行过程中，存在着很多的不确定因素：输入数据的不确定性（任务到达间隔时间）、过程模型结构的不确定性（分支概率）、流程对象属性的不确定性（活动执行时间）等。

通过上述分析可见，目前 XPDL 所带的仿真信息只能设置固定的参数，对仿真过程中的一些不确定性描述还不够，根据实际的需求，按照第二章定义的面向仿真的工作流过程定义元模型，我们需要对 XPDL 进行扩展，扩展后的过程

定义语言我们称之为 SIM-XPDL。

3.4 SIM-XPDL 的语法规范

3.4.1 SIM-XPDL 设计原则

1. 对 XPDL 的兼容性

SIM-XPDL 的设计将不修改 XPDL 中已有的定义，同时也不会修改其旧有的语义。这意味着企业中原有业务流程的 XPDL 模型在当前新 SIM-XPDL 的 XML 模式下仍保持着有效性；SIM-XPDL 继承了 XPDL 中的所有元素，即 SIM-XPDL 是 XPDL 的一个子类。

2. 高内聚性

在 XPDL 中已考虑了许多复杂的应用场景需求，所以 SIM-XPDL 不应过多的对非涉及仿真建模需求的元素进行设计（除了一些专门为了弥补 XPDL 不足的元素之外）。即 SIM-XPDL 所设计的新的建模元素应该是高内聚的，不应该涉及 XPDL 模型的过多元素的扩展。否则，SIM-XPDL 将与原则 1 发生冲突，因为随着 SIM-XPDL 复杂度的增加，相应对 SIM-XPDL 的解析成本会增加，同时，也加大了开发支持 SIM-XPDL 的仿真引擎的难度。

3. 可扩展性

SIM-XPDL 应该继续提供 XPDL 中扩展属性的支持机制，同时，在对仿真的支持方面，在尽可能满足当前已知的仿真应用场景需求的同时，仍应提供相关的扩展元素来满足将来业务用户的复杂需求，从而可以在不修改 SIM-XPDL 的 XML 模式和引擎内核的情况下快速响应用户需求。

3.4.2 XPDL 的扩展

根据上述的原则，我们根据仿真过程中的不确定性对活动执行时间和分支概率进行定义。由于任务到达时间可以放在事务模型中进行设置，因此在这里没有考虑进行该元素的定义。

1. 活动执行时间的定义

工作流模型执行过程中活动从被激活到被完成分为接受时间和执行时间两个部分，其中接受时间是从该活动被激活到有资源来接受的这段时间间隔，执行时间是从该活动被接受到被完成的时间间隔。在原有仿真信息的定义中，包括了活动的等待时间（WaitingTime）和工作时间（WorkingTime）以及预计活动持续时间（Duration）的定义，这些时间的定义只能是设定固定的时间，因此在本文中没有运用原有的这些仿真时间的定义，对于等待时间本文根据仿真过程中资源

等待的情况来记录。在本章中着重对活动的执行时间进行了定义，使之可以支持动态的执行时间，为区别于原有的工作时间（WorkingTime）的定义，其扩展定义为 WorkTime。我们将活动工作时间分为两种类型，一种是固定时间另一种是动态时间，动态的时间是随机的，也许会遵循某个特定的分布函数，因此我们对工作流仿真模型中活动的执行时间（WorkTime）的定义如下：

```
<xsd:element name="WorkTime">
  <xsd:complexType>
    <xsd:choice>
      <xsd:element ref="xpdl:FixedTime" minOccurs="0"/>
      <xsd:element ref="xpdl:DynamicTime" minOccurs="0"/>
      <xsd:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
    </xsd:choice>
  </xsd:complexType>
</xsd:element>
```

以下是固定时间（FixedTime）的定义，其单位包括秒、分钟、小时、日、月和年。

```
<xsd:element name="FixedTime">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element ref="xpdl:TimeValue" minOccurs="0"/>
      <xsd:any namespace="##other" processContents="lax" minOccurs ="0"
       maxOccurs="unbounded"/>
    </xsd:sequence>
    <xsd:attribute name="TimeUnit">
      <xsd:simpleType>
        <xsd:restriction base="xsd:NMTOKEN">
          <xsd:enumeration value="Y"/>
          <xsd:enumeration value="M"/>
          <xsd:enumeration value="D"/>
          <xsd:enumeration value="h"/>
          <xsd:enumeration value="m"/>
          <xsd:enumeration value="s"/>
        </xsd:restriction>
```

```
            </xsd:simpleType>
          </xsd:attribute>
          <xsd:anyAttribute namespace="##other" processContents="lax"/>
        </xsd:complexType>
      </xsd:element>
      <xsd:element name="TimeValue">
        <xsd:complexType>
          <xsd:simpleContent>
            <xsd:extension base="xsd:string">
            <xsd:anyAttribute namespace="##other" processContents ="lax"/>
            </xsd:extension>
          </xsd:simpleContent>
        </xsd:complexType>
      </xsd:element>
```

动态执行时间（DynamicTime）的定义，根据实际的需要我们给出了常见分布的定义，包括泊松分布（Possion），正态分布（Normal），指数分布（Exponet），均匀分布（Uniform）。

```
      <xsd:element name="DynamicTime">
        <xsd:complexType>
          <xsd:choice>
            <xsd:element ref="xpdl:Possion" minOccurs="0"/>
            <xsd:element ref="xpdl:Normal" minOccurs="0"/>
            <xsd:element ref="xpdl:Exponet" minOccurs="0"/>
            <xsd:element ref="xpdl:Uniform" minOccurs="0"/>
            <xsd:any namespace="##other" processContents="lax" minOccurs= "0"
            maxOccurs="unbounded"/>
          </xsd:choice>
        </xsd:complexType>
      </xsd:element>
```

对于常见分布的定义这里就不再详细描述。设计好的活动执行时间的schema 在 XML 编辑软件 XMLSpy 中图形化显示如图 3-1 所示。

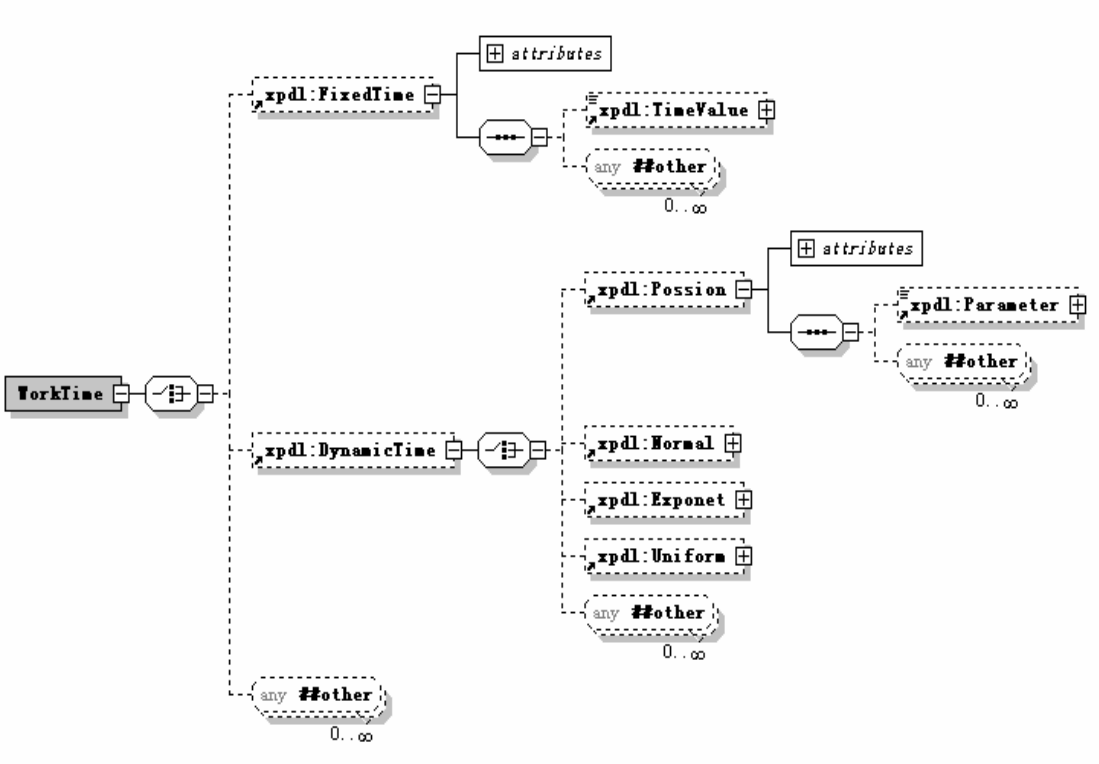

图 3-1 WorkTime 元素 XML-schema 结构图

2. 转移的扩展定义

活动间是通过转移相互联系起来的，每个转移都有三个基本的属性：源活动（From）、目标活动（To）和转移条件（Condition），其中转移条件可以为空。XPDL 中转移的转入类型分为 XOR-Join 和 AND-Join，转出转移分为 AND-Split 和 XOR-Split。从一个活动到另一个活动的转移可能需要转移条件用来判断转移是否可以进行；转移可能产生一个连续的活动序列，也可能产生一系列活动的并行执行。复杂的转移是不能通过上述提到的通过上述的几个基本属性及离开、进入动作表示清楚的，有时候还需要依靠虚活动（root）来构造。虚活动作为一些活动的中间步骤，可以把一些离开动作和进入动作连接在一起，使用带有虚活动的基本转移实体可以实现复杂的路由结构。

对于转移条件为空的 AND 型分支工作流的处理是系统会执行所有转移指向的目标活动；如果 AND 型分支的转移带有条件则根据是否满足条件来选择下一步要执行的活动，这些转移条件的判断和计算是同时进行的，与源活动中转移列表的顺序无关，这种情况可以看作为多重选择（Multiple Choice）。对于 XOR-Split，系统是根据转移上的条件来选择执行的后续活动。选择哪条转移路径需要根据每个转移上的转移条件来决定，按照转移在列表中的顺序来对转移条件进行逐个判断或计算，满足则执行，不满足则执行下一个，如果计算到一个无条件转移或者一个转移的条件是 OTHERWISE 类型，那么结束对列表中转移的判断。XOR-Split

只要有一个转移满足转移上的条件就直接执行该转移，并结束判断，不管在列表中是否还有其它满足条件的转移。

在仿真过程中我们必须要自动地给出路由的选择，对于有选择判断的转出转移属于决策型输出，如带有条件的 AND-Split，XOR-Split 等，它们会根据转移上的条件来执行，其选择的结果是随机的，对于每条可能的输出分支都具有一个概率[14]。

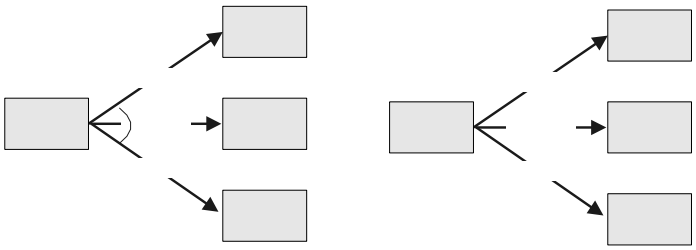

图 3-2 AND-Split 和 XOR-Split 示例

如图 3-2 所示，对于 XOR-Split，根据条件它可能选择执行活动 a2，也可能选择执行活动 a3 或 a4。对 AND-Split 会并发执行满足条件的转移指向的活动，因此我们需要提供对选择性分支概率的描述，这里我们可以充分利用 XPDL 中 Transition 的扩展属性，在不改变 XPDL 原有框架的基础上提供对选择性分支概率的描述。基于语义的需要我们在扩展属性中加入描述随机转移被选择执行的概率的属性“SplitProbability”。“SplitProbalility”的值表示该转移被选择执行的概率。在缺乏先验知识的情况下，其概率由用户自行给出，给出示例如图 3-3 所示。

AND-Split a2 a1 condition1 condition2 a3 condition3 a4

```
<Transition>
        …
   <ExtendedAttributes>
        …
      <ExtendedAttribute Name="SplitProbability" Value="0.0,0.32"/>
   </ExtendedAttributes>
</Transition>
```

图 3-3 转移概率示例

对于一般型的转出转移我们不需要对该属性进行定义，决策型的转出转移就需要我们手工的添加。上述定义表示该 Trasition 被选择执行的概率为 0.32（总概率和为 1）。

3.5 SIM-XPDL 的验证及兼容性分析

建模的正确性是业务目标实现的基本保证。检测模型设计的正确在工作流模型仿真不可缺少的重要前提。通过对工作流模型的验证机制可以保证过程定义的

语法和结构正确，从而使仿真能够顺利进行。一般模型检验模块是集成在建模工具中，因为本文没有提供对扩展的仿真信息可视化的编辑界面，需要手工添加，因此对于该 schema 的结构验证和有效性我们首先在 XMLSpy 下进行，逻辑及其它验证我们沿用原有的验证机制在建模工具或工作流引擎中进行。Java 环境下的 SIM-XPDL 文档解析验证在第五章有具体实现的部分详细介绍。

表 3-1 XPDL 与 SIM-XPDL 的语言特征对比

语言特征	XPDL	SIM-XPDL
图语言否	是	是
人工交互	支持	支持
异常处理	很弱支持	同 XPDL
顺序结构	支持	同 XPDL
并发结构	AND-Split, AND-Join	同 XPDL
选择结构	XOR-Split, XOR-Join	同 XPDL
循环结构	支持	同 XPDL
时间控制	Deadline 和 Limit	同 XPDL
成本参数	支持	同 XPDL
实例化次数	支持	同 XPDL
静态工作时间参数	支持	支持
动态工作时间参数	不支持	支持
转移分支概率参数	不支持	支持
静态等待时间	支持	同 XPDL

根据 FOLDOC[46]上对兼容性的定义，分别从 SIM-XPDL 的向前和向后兼容两个方面进行分析。兼容性意味着新旧版本的协调，根据我们设计的原则，SIM-XPDL 继承了 XPDL 所有的元素，对于原有的 XPDL 模型在支持 SIM-XPDL 模型的系统中仍然是有效的，因此，我们说是满足向后兼容的。通过如下给定的例子来说明向后兼容性，使用 sim_xpdl.xsd 文件对未进行扩展前的 XPDL 实例文档进行验证。为了能够使用我们定义的 schema 文档，我们需要将 xsi:schemaLocation 指定为本地的 schema 文件：

xsi:schemaLocation="… f:/sim_xpdl.xsd"

指定的 schema 为本地 F 盘下的 sim_xpdl.xsd 文件。通过测试证明原来的 XPDL 模型可以通过有效性验证。

向前兼容比向后兼容要难以达到，因为需要考虑到将来的发展情况。在原来定义的 XPDL schema 中就考虑到了这样的问题，因此提供了扩展属性以及<any>

元素可以使我们在 XML 文档中添加没有被 schema 定义过的新元素从而扩充 XML 文档[47]。在对 XPDL 进行扩展的时候也充分考虑了向前兼容的情况，通过 <xsd:any namespace="##other" processContents="lax" minOccurs= "0" maxOccurs ="unbounded"/>定义语句使得扩展后的 SIM-XPDL 也具有相应的扩展性。其中，XPDL 与 SIM-XPDL 的语言特征对比如表 3-1 所示。

3.6 小结

本章对现有的工作流过程定义语言进行了分析，工作流管理联盟的过程定义语言 XPDL 提供了基础的仿真信息，但不能满足我们目前工作流模型仿真的需求。按照第二章提出的支持仿真的工作流过程定义元模型、针对仿真过程中的不确定性、依据兼容性、高内聚性、可扩展性原则对 XPDL 的扩展，使之可以支持仿真过程中的不确定性描述。通过这种方法对 XPDL 进行扩展没有改变原有的建模元素，所以不会对原有的工作流模型的功能和结构进行修改，从这个方面也保证了仿真模型和原有工作流模型在结构和功能上的一致性。

第四章 工作流仿真环境参数设置

在第二章我们提到过执行仿真之前我们需要定义仿真场景，场景就是一组与仿真运行相关的数据。仿真执行之前我们需要设定的仿真数据除了我们在第三章中定义的活动执行时间之外还包括仿真起始时间设置、事务生成器和时间表定义，我们将其统称为仿真环境。在仿真过程中仿真可信度是非常重要的，如果仿真模型不能反映实际的业务流程，那么仿真的结果和分析是没有意义的。本章主要解决的问题就是将工作流的过程实例历史数据作为观测数据进行统计和分析之后得到仿真运行相关数据的参考设置，从而提高仿真结果的可信度和有效性。上述这种方法适用于已存在的工作流模型，对于新建的工作流模型我们往往需要依靠专家经验等作为仿真参数的参考设置。

4.1 工作流仿真环境概述

工作流模型仿真的过程第一步是设置仿真环境，然后是执行仿真。仿真环境符合实际的情况是仿真结果可信的前提，在我们的仿真系统中需要设置的仿真环境主要包括仿真运行设置、事务生成器设置、事件表定义和其他的仿真选项[48]。

其中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，比如，定义仿真一次仿真运行的开始时间是 2007 年 04 月 20 日，定义结束时间是 2007 年 05 月 20 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。当时间到达结束时间时仿真就终止。

事务生成器的设置会对仿真结果产生重要影响，根据第二章中对事务模型的定义，事务生成器需要提供完成型、间隔型和需求型三种类型的事务设置。

在实际的业务系统中，资源的使用会遵循一定的工作时间表。时间表定义需要根据实际的流程情况为资源定义工作时间表，确定资源可以被利用的时间区间。除了设置上述的条件外，有时候要根据不同的仿真软件的具体要求，设置其他的仿真选项。

4.2 基于工作流历史数据分析的仿真环境参数设置

4.2.1 历史数据的特点

工作流模型仿真属于离散事件仿真系统，离散事件系统一般会有一个以上的随机变量。如在单服务台系统中[49][50]，顾客的到达时间是随机变量，服务员为

顾客服务的时间也是随机变量。类似地，对于工作流管理系统来说，过程的启动，活动的激活、处理时间以及过程的分支选择等也都是随机的。对具有随机变量的系统，首先需要确定其随机变量的分布类型及其参数，以便仿真运行时根据这些分布和参数产生随机变量。确定随机变量的分布类型及其参数的基础是收集该随机变量的观测数据。

在工作流日志中包含了实时数据和历史数据，综合了包括过程、组织、资源、信息等大量工作流运行状态信息，这些对工作流的性能分析是非常有价值的[51]。实时数据是指当前工作流系统中仍处于激活状态的、未完成的过程实例所产生的数据和信息，历史数据是指已完成的过程实例所产生的一系列的数据和信息。工作流日志的分析是近几年工作流研究的新热点，并且已取得了一些研究成果。工作流的建模是一个复杂和耗时的工作，Van der Aalst[52]提出了从工作流日志中挖掘工作流模型的机制，通过算法从日志中提取过程模型并以 Petri 网的形式表达出来；Onoda[53]也提出了通过分析执行日志来改进工作流模型的参数，并且使用仿真的方法来分析系统的瓶颈；Eder[54]提出了采用数据仓库和 OLAP（联机分析处理）来进行工作流日志分析。文献[55]利用执行成功的过程实例的历史信息作为样本统计出更符合实际运行情况的属性值和指标值，使用这些值自动的改进工作流过程定义。在此基础上，提出一种工作流过程改进的模型，模型主要有实例提取器，学习器和改进器组成，利用各自的规则，实现工作流过程改进。可以看出，工作流历史数据有助于日后查询、分析与报表统计，同时也可作为决策支持以及过程定义改进辅助的参考数据。虽然历史数据在时效上已不重要，但对整个系统的全局分析及改进却极其重要。结合工作流历史数据的特点及前人的工作，我们将工作流历史数据作为观测数据不乏为一种可行的方法。通过对我们所需设置的仿真参数的分析，得出我们需要从历史数据中提取的信息主要包括过程的启动间隔时间，活动完成时间等数据信息。

4.2.2 历史数据的迁移和备份

工作流的数据库包括模型库和实例库，其中历史数据和实时数据存放在实例库中。随着时间的推移，库中的数据记录会越来越多，会对工作流管理系统的整个性能造成影响。因此将这些不经常查询或适用的数据进行备份转储，既可以对工作流管理系统的运行性能有所提高，也可以通过对历史数据的备份和处理方便我们对历史数据进行分析和处理的需求。例如，对于一个月之前已完成的过程实例数据我们往往很少再去使用或查询，我们可以将超过某个期限的已完成的过程实例的相关数据记录转储备份出来。如图 4-1 所示，将已经结束的过程实例的相关数据从工作流数据库中提取出来，扩展为模型库，实例库和备份库。

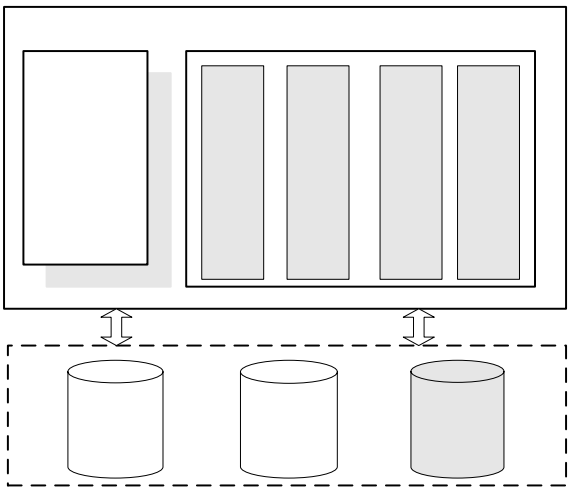

图 4-1 工作流历史数据管理模块

图 4-1 中的灰色部分是为了解决历史数据备份的问题添加的功能模块。备份库用于存放从实例库中迁移出来的历史数据，其中备份库的库表结构与原工作流系统保持一致，方便对备份数据的访问。历史数据管理器包括历史数据迁移模块、实时同步控制模块、历史数据监控模块和历史数据分析模块，其中历史数据迁移模块负责完成工作流的过程实例历史数据从实例库向备份库的迁移行为，“自动迁移”的时机由一个监视线程来完成，这个监控线程由历史数据监控模块提供，它将进行迁移时间限制（深夜或凌晨时间）和备份周期条件（每周一次或每月一次）的检查，只有在两者条件均满足的情况下，才会自动执行；实时同步控制模块可以让用户选择以获取最新的过程实例历史信息；历史数据分析模块主要实现对工作流过程实例的历史数据的分析功能，如进行历史数据的查询，历史数据的挖掘分析等。

通过历史数据管理器解决了工作流管理系统的性能问题，并且对于我们利用工作流历史数据进行仿真环境的分析研究也提供了便利。

4.2.3 历史数据的提取

历史数据管理器提供了工作流历史数据的抽取接口，对于这个数据抽取的接口我们可以借鉴文献[55]的思想，在基础上提出更适合我们的一系列的规则：

1. 提取指定的信息，主要根据指定标识提取相应的实例信息，如过程实例ID；

2. 提取实例的数目，可以根据实际的要求来选择提取实例的数目，也可以随着实例数的多少而改变。如用户可以指定提取某个过程的活动在一段时间内的执行数据；

3. 抽取的信息并不是所有的属于该指定标识的信息，我们可以有选择的提

取，对抽取后的信息会根据一定的过滤规则，去除不完整的、含噪音节点的和重复的历史信息。

4.3 仿真环境参数的确定

4.3.1 确定仿真环境参数的实现步骤

为了获得正确的仿真运行参数数据，根据第二章的历史数据模型，给出历史数据标识 id 以及指定标识的历史数据的属性集 r，根据 id 和 r 组装查询历史数据的语句，通过如下步骤可以获得历史数据的值域 v 及分析的函数和值 f。根据离散事件系统对仿真输入数据及随机变量确定的步骤，本文确定仿真环境参数的具体步骤如下：

Step1 给定需要获取数据的标识，调用历史数据管理器 getHistoryData (String queryString)接口，获取所需数据；

Step2 进行分析前的预处理；

Step3 根据观测数据做出直方图或线图，假设观测数据的分布函数类型；

Step4 对分布函数的参数估计；

Step5 拟合优度检验，判断假设是否成立。如果假设成立就直接给出其分布类型及参数，结束判断。如果收集到的数据和假设的分布形式不相符合，则返回到 Step3，并给出一个不同的假设，重复上述过程，若重复多次仍不符合，则跳到 Step6；

Step6 根据实际情况，给出经验分布形式。

通过上述步骤最终可以确定某个随机变量的分布函数类型，其结果可应用于仿真环境的设置。

1. 观测数据的收集

在解决实际问题的时候，数据的收集是工作量较大的一项工作。如果收集的数据不正确，利用这样的数据作为决策的依据必将导致错误，造成损失和浪费。在收集数据时我们需要考虑以下两点：

做好计划从观察入手，收集需要分析的相关数据；

在收集数据时要注意分析数据，只收集与仿真需要相关的数据，对仿真无用的数据就没有必要多加收集。

2. 分布类型的假设

由观测数据来确定随机变量的分布类型，最常用的方法是对观测数据进行适当的预处理，然后根据预处理的结果对分布类型进行假设。

最常用的预处理方法分别是点统计法，直方图法、概率图法和线图法[49][50]。

其中直方图法将观测数据 $x_1, x_2, ..., x_n$ 的取值范围分成 k 个断开的相邻区间 $[b_0, b_1), [b_1, b_2), ..., [b_{k-1}, b_k)$，每个区间宽度相等，记为 $\Delta b = b_j - b_{j-1} (j = 1, 2, ..., k)$。对任意 j，令 g_j 表示第 j 个区间 $[b_{j-1}, b_j)$ 上观测数据的个数占整个观测数据的比例数，即设 n_j 为第 j 个区间上观测点的个数，则

$$g_j = n_j / n (j = 1, 2, ..., k) \qquad \text{公式（4-1）}$$

定义函数

$$h(x) = \begin{cases} 0 & x < b_0 \\ g_j & b_{j-1} \le x \le b_j \\ 0 & b_k \le x \end{cases} \qquad \text{公式（4-2）}$$

做出 $h(x)$ 的直方图，再将该图与常见理论分布的密度函数图形进行比较（先忽略位置及比例尺的差别），观察何种分布与 $h(x)$ 的图形类似，则可假设观测数据 $x_1, x_2, ..., x_n$ 服从该类型分布。在实际使用时，对于数据量较少的情况下，可能需要对观测数据进行适当的调整，即增加一些其值特别大或特别小的观测数据，以便与理论分布进行比较，使用直方图的困难在于如何确定区间长度 Δb。Δb 太长，将丢失信息；Δb 太小，则观测数据中的噪声虑除得不够。我们可以采用一种确定组数和组距的经验公式，这一公式是统计学家斯特杰斯（H.A.Sturges）创建的，称为斯特杰斯经验公式，即：

$$k = 1 + 3.322 \lg n \qquad \text{公式（4-3）}$$

由此推出

$$\Delta b = (x_{\max} - x_{\min}) / k \qquad \text{公式（4-4）}$$

线图法首先需要将观测数据 $x_1, x_2, ..., x_n$ 将其按递增顺序排列，设共有 m 个取值（$m \le n$），分别为 $x(1), x(2), ..., x(m)$。记 $x(i)$ 的数据个数 f_i 占整个观测数据个数的比例为 h_i，即

$$h_i = f_i / n (i = 1, 2, ..., m) \qquad \text{公式（4-5）}$$

线图法是将观测数据的线图和假设的理论分布的质量函数进行比较以便找到相似的图形。可以看出，线图法不需要将观测数据划分新的区间，因而不会丢失任何信息，比起直方图法使用起来更为方便。

对于分布类型的假设，我们根据从历史数据库提取出来的信息，通过统计分析软件及 Matlab 进行预处理，根据上述方法对所获得的数据进行分析得出其分布类型。

3. 参数估计

在我们做出分布的假设之后，下一步便是分布参数的估计。

（1） 样本均值和样本方差

对观测数据 $x_1, x_2, ..., x_n$，其样本均值 $\bar{X}$ 定义为

$$\bar{X} = \frac{\sum_{i=1}^{n} x_i}{n} \qquad \text{公式（4-6）}$$

样本方差为

$$S^2 = \frac{\sum_{i=1}^{n} x_i^2 - n\bar{X}^2}{n-1} \qquad \text{公式（4-7）}$$

如果离散数据已按频数分组，则

$$\bar{X} = \frac{\sum_{i=1}^{k} f_i x_i}{n} \qquad \text{公式（4-8）}$$

$$S^2 = \frac{\sum_{i=1}^{k} f_i x_i^2 - n\bar{X}^2}{n-1} \qquad \text{公式（4-9）}$$

其中 k 为分组数， f_i 为数值 x_i 的观测频数。

（2） 估计量

由观测数据估计某一分布参数的方法很多，使用最普遍的是最大似然估计，其他还有最小二乘估计、无偏估计等。对于常见的分布函数参数的最大似然估计我们可以参考相关资料。

4. 卡方检验

由观测数据假设了分布函数的类型之后，一般需要检验该分布与这些观测数据吻合的程度，即进行拟合优度检验。对拟合分布进行 χ^2 检验时，要将该拟合分布的取值范围分为 k 个相邻区间 $[a_0, a_1), [a_1, a_2), ..., [a_{k-1}, a_k)$，然后计算

$$P_j = \sum_{\{I: a_{j-1} \le x_i < a_j\}} \hat{P}(x_i)(j = 1, 2, ..., k) \qquad \text{公式（4-10）}$$

或

$$P_j = \int_{a_{j-1}}^{a_j} \hat{f}(x)dx(j = 1, 2, ..., k) \qquad \text{公式（4-11）}$$

公式（4-10）、（4-11）分别对应于离散和连续的情形，其中 $\hat{P}(x_i)$（ $x_1, x_2, ..., x_n$ ）是离散情况下拟合分布的质量函数， $\hat{f}(x)$ 是连续情况下拟合的分布密度函数。为了使检验无偏，应该使每个区间的 P_j 尽可能相等，并能使 $nP_j \ge 5$。计算 χ^2 检验的统计值：

$$\chi^2 = \sum_{j=1}^{k} \frac{(N_j - nP_j)^2}{nP_j} \qquad \text{公式（4-12）}$$

其中 $N_j(j=1,2,...,k)$ 为每个区间上观测数据的个数，$nP_j(j=1,2,...,k)$ 是拟合分布得到的期望个数。给出检验水平 α， 判断准则有两种：

（1） 若 $\chi^2 \geq \chi_\alpha^2(k-1)$，则拒绝如下原假设 H_0：观测数据 x_i 是密度函数为 $\hat{f}(x)$ 或质量函数为 $\hat{P}(x)$ 的独立同分布随机变量；否则不拒绝 H_0。

（2） 若 $\chi^2 \leq \chi_\alpha^2(k-r-1)$，则不拒绝原假设 H_0，否则拒绝 H_0，其中 r 是拟合分布估计参数的个数。在 k 较大（如 $k>10$）且 $r \leq 2$ 时，较多采用第一种准则。

4.3.2 实例分析

运用上述的方法，我们对政务审批系统中公安局通用流程的受理活动完成时间进行分析，其过程定义在建模工具 JaWE 下的图形化表示如图 4-2。

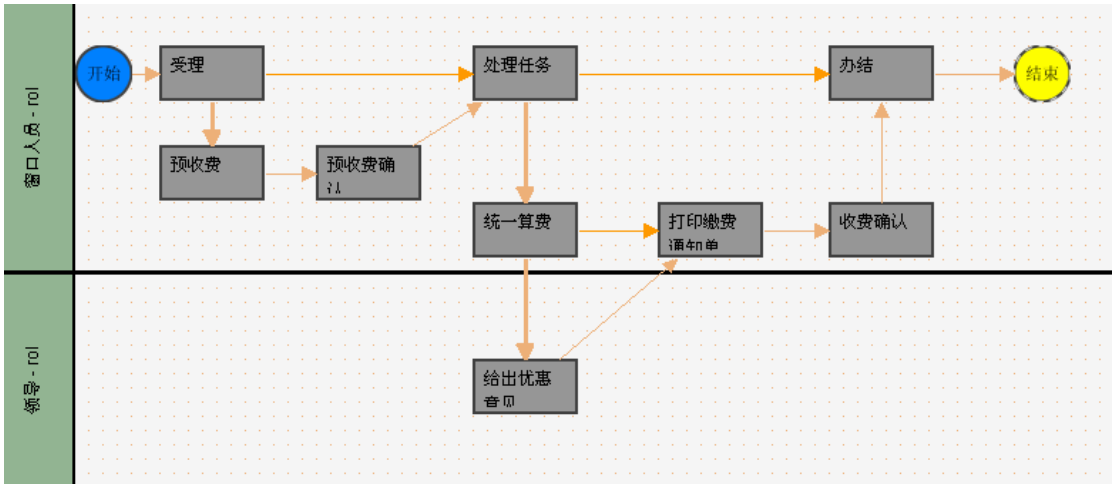

图 4-2 公安局通用流程在建模工具 JaWE 中的图形化表示

首先从备份库中提取出我们需要的数据，其中获得的活动完成时间的样本数据如表 4-1 所示。

表 4-1 活动完成时间（时间单位：ms）

完成时间	频数	完成时间	频数	完成时间	频数
[0,1000)	598	[6000,7000)	13	[12000,13000)	3
[1000,2000)	22	[7000,8000)	4	[13000,14000)	1
[2000,3000)	69	[8000,9000)	3	[14000,15000)	0
[3000,4000)	50	[9000,10000)	4	[15000,16000)	1
[4000,5000)	20	[10000,11000)	4		
[5000,6000)	19	[11000,12000)	1		

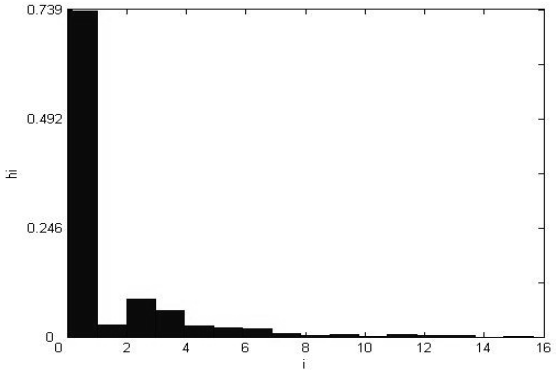

图 4-3 观测数据直方图

表 4-2 活动完成时间模型的卡方检验

j	区间	N_j	$(N_j-nP_j)^2/(nP_j)$
1	[0, 60.29)	8	0.065
2	[60.29, 123.85)	431	9.384
3	[123.85, 191.03)	35	0.002
4	[191.03,262.30)	86	0.127
5	[262.30,338.16)	17	0.034
6	[338.16,419.26)	8	0.065
7	[419.26,506.37)	7	0.070
8	[506.37,600.45)	3	0.087
9	[600.45,702.73)	0	0.102
10	[702.73,814.77)	1	0.097
11	[814.77,938.61)	2	0.092
12	[938.61,1077.06)	0	0.102
13	[1077.06,1234.02)	0	0.102
14	[1234.02,1415.22)	2	0.092
15	[1415.22,1629.53)	0	0.102
16	[1629.53,1891.83)	6	0.074
17	[1891.83,2229.99)	27	0.011
18	[2229.99,2706.59)	30	0.007
19	[2706.59,3521.36)	59	0.021
20	[3521.36,∞)	90	0.150

图4-3的形状与指数分布的密度函数比较接近，假设观测数据服从指数分布，其中 $\lambda=\overline{x}(812)=\sum_{i=1}^{n}x_i/812=1175.458$ 。为了保证 $nP_j\geq 5$，取 $P_j=0.05$，$k=20$，确定区间后，分别得到 N_j 和 $(N_j-nP_j)^2/(nP_j)$，如表 4-2 所示。

最后计算 χ^2 值看假设是否成立，$\chi^2 = 10.875$。给定的检验水平 $\alpha = 0.05$，查 $\chi^2_{0.05}(19) = 30.144$。因为 $\chi^2_{0.05}(19) > \chi^2$，所以不拒绝假设，因此我们设定该活动完成时间服从 $\lambda = 1175.458$ 的指数分布。

4.4 小结

本章将工作流的历史数据通过添加的功能模块从实例库迁移到备份库，并对数据进行分析处理作为仿真环境的参考设置，这样可以很好地解决工作流历史数据量较大时工作流系统性能的问题，并且为我们从历史数据中抽取信息提供了很大的方便。在仿真中可信度是一个非常重要的问题，仿真参数的设置是否符合实际情况是仿真结果可信的重要前提。在工作流管理系统中，工作流的历史数据详实地记录了工作流模型的运行轨迹和状态，我们利用这些数据的分析结果可以作为仿真参数的参考设置。

第五章 工作流模型仿真系统

本章描述了工作流模型仿真系统的体系结构及其主要的相关组件。工作流模型仿真系统的设计以仿真引擎为核心，该仿真引擎通过调用工作流引擎接口模拟用户执行工作流实例，其中业务过程采用 SIM-XPDL 语言描述，添加了仿真时钟、随机变量生成器等部件实现工作流模型的仿真。

5.1 工作流模型仿真系统设计

5.1.1 系统设计目标

工作流模型的仿真是业务过程在建模之后投入实际使用之前的模拟执行过程，可以看作是对业务过程建模后的一个测试过程。工作流模型仿真系统从建模到执行阶段需要提供包括工作流过程模型的建模工具、工作流引擎、仿真引擎以及数据库的支持。工作流模型仿真系统需要能够模拟工作流模型的执行、模拟用户触发工作流过程实例并自动执行其工作流列表中的工作项。在工作流引擎中提供了一系列的接口函数，因此我们希望可以在此基础上构建仿真引擎。其中对于工作流模型的仿真可以分为业务过程的性能仿真、功能仿真和结构仿真三种。性能仿真对业务过程分析执行时间、成本等性能，功能仿真可以判断过程模型是否实现了指定的业务过程功能，结构仿真主要分析仿真执行时的过程执行路径。

5.1.2 系统总体结构

鉴于工作流执行机制与仿真机制的相似性，本文采用以工作流引擎为基础构建仿真引擎来实现工作流模型仿真系统（WMSS）。WMSS 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用户，对应于建模阶段、仿真环境设置和运行阶段分别给出其组件。

建模阶段：工作流过程定义工具（JaWE），对于其仿真信息的设定，本文没有提供图形化的设置界面需要人工手动添加。

仿真环境设置：事务生成器设置、时间表、资源、组织结构以及应用程序的设置和映射、历史数据管理器等。

运行阶段：工作流引擎、SIM-XPDL 解析器、资源池、仿真时钟和随机变量生成器等。

WMSS 的支撑系统包括数据库系统（Oracle）和 Ehydra DODS 等。整个系统的组件采用面向对象的方法设计，主要的系统实现语言为 Java 语言。

如图 5-1 所示，工作流模型仿真系统分为三个部分，分别由用户界面层、业务逻辑层和数据库层组成。其中用户界面层提供了对仿真起止时间、事务生成器进行设置以及仿真分析报告的功能。

业务逻辑层包括了工作流引擎和仿真引擎两大部件，仿真引擎通过提供的标准化的 API 和交换接口调用工作流引擎的功能从而模拟用户实现仿真，其中仿真引擎包括了仿真时钟、随机变量生成器及 SIM-XPDL 解析器，这几个部件为仿真基础部件，提供仿真引擎执行的基础支持。历史数据管理器用于对工作流历史数据的挖掘和分析，同时这个组件也可用于对仿真数据的分析。资源池主要用于存放仿真时所需的资源。

数据库层包括了工作流模型库、工作流实例库、工作流历史数据库、工作流仿真数据库。对于所有持久数据均可在 Enhydra 的开源 DODS 工具的支持下实现 O/R Mapping。

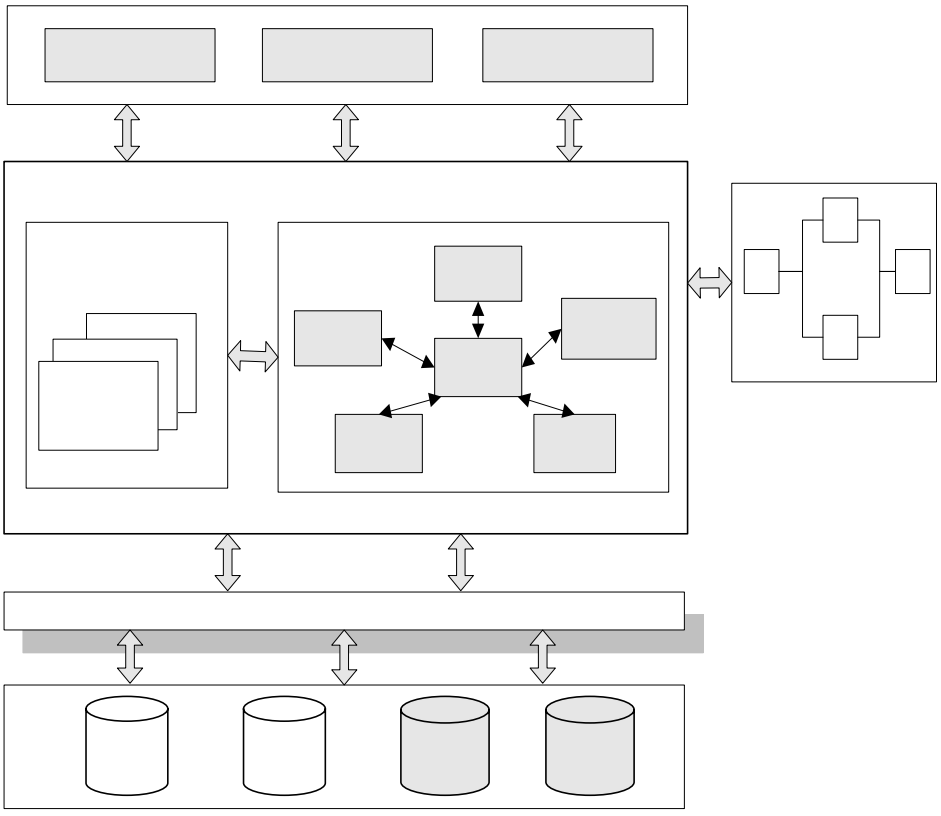

图 5-1 工作流模型仿真系统总体结构

5.2 系统的关键组件及其实现

5.2.1 工作流引擎

通过对多个主流的开源工作流引擎的对比，本文决定选择开源工作流引擎

Shark[56]。Shark 是一个由国际组织 Enhydra 开发的开源工作流研究项目，是遵循 WfMC 的 XPDL 标准的 Java 开源工作流引擎。在开源工作流引擎中，Shark 完全采用标准模块化设计，提供标准接口以方便系统改进和功能扩展。它已成为一个最符合 WfMC 规范、具有较高学术研究价值和商业应用价值的 WfMS 范例。Shark 主要由工作流引擎管理器、Package 管理器、Process 管理器、Activity 管理器、EventAudit 管理器、持久层服务和日志管理器等功能模块组成。通过 ToolAgent 管理器可以调用外部应用如 EJB、Java Class、JavaScript、WebService、Corba 等。

工作流执行引擎和仿真引擎的一个差别就是前者和外界存在实际的交互，而后者与外界的交互都是虚拟的，并不是产生真正的互动作用。了解这两点不同以后我们就可以最大限度地重用工作流引擎的执行机制。仿真引擎通过调用工作流引擎提供的接口，模拟用户执行工作项，从而达到自动推进工作流过程实例自动的向前推进。其中 WAPI 是一组工作流应用程序接口函数及相应的数据交换格式，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，其中仿真引擎主要完成的模拟用户的交互功能主要是通过接口 2 实现的。接口 2 主要定义了对过程模型定义操作、过程实例管理功能、过程状态管理功能、过程状态管理功能、任务项处理功能、过程监控功能及其它管理功能等。

5.2.2 SIM-XPDL 解析器

在对 WfMC 的过程定义语言标准 XPDL 扩展后，存在 Shark 无法识别的元素，Shark 缺乏对这些概念扩展的正确性校验机制。因此 SIM-XPDL 解析器的主要工作包括扩展元素的解析，其中对模型的校验也是很重要的一个功能。

1. 模型的校验

包（Package）是 XPDL 元素实体中的根节点实体，由过程、参与者、相关数据以及外部应用定义等组成，同时包之间可以相互引用。

由于包是一个 XML 实体，因此在校验之前先进行 XML 的结构校验以及合法性校验。XML 的编程接口分为两大类，文档对象模型（DOM，Document Object Model）和基于事件的 SAX（Simple API for XML）接口。前者将解析整个 XML 树并保存在内存中，而 SAX 则每当发现一个新的元素时产生一个报告事件。由于 Shark 会反复的用到包的 XML 树，因此选用了 DOM 接口。

原有 Shark 的包校验由类 org.enhydra.shark.xml.PackageValidator 完成，包校验接口的使用方法如图 5-2 所示。XMLInterfaceForJDK13 首先利用 DOM 接口进行包的 XML 解析，其 parseDocument 方法将从包的 DOM 树中提取各个节点内容生成 Shark 中为 XPDL 各个元素定义的类实例。PackageValidator 接收 Package 对象以及其他相关参数初始化。最后调用其 validateAll 方法进行校验，该方法返

回一个布尔变量，true 为正常返回，false 表示发生了错误。校验的错误信息将存放在 Map 类型的 parsingErrorMessages 中。

```
XMLInterfaceForJDK13 xmlI=new XMLInterfaceForJDK13();
Package pkg=xmlI.parseDocument(args[0],true);
PackageValidator validator = new PackageValidator(pkg,false, true,false, true,
false,false,"UTF-8");
validator.validateAll(false);
```

图 5-2 包校验部分代码

我们采用的是改进后的 schema，其中 XMLInterfaceForJDK13 中的 parseDocument()方法中需要对 schema 进行指定，图 5-3 中描述了设定指定 schema 的部分代码。

```
DOMParser parser = new DOMParser();
File schema=new File("F:\\sim-xpdl.xsd");
parser.setProperty("http://java.sun.com/xml/jaxp/properties/schemaSource",
schema);
parser.parse(new InputSource(new FileInputStream(f)));
```

图 5-3 通过指定的 schema 进行校验的部分代码

2. 模型的解析

在执行仿真过程中，活动执行时间、转移概率的解析是仿真引擎推进工作流过程实例向前推进的重要的信息，因此 SIM-XPDL 解析器需要从过程定义中读取并解析执行时间及转移的相关仿真信息。

工作流模型仿真系统采用面向对象的设计开发方法，对于建模工具提供的模型元素都有一个相应的类，根据第三章 SIM-XPDL 中添加的仿真元素，在仿真模型执行过程中为了获取仿真信息，在 org.enhydra.shark.xpdl.elements 包下对添加的仿真信息节点建立相应的 Java 类。扩展后的 SimulationInformation 包含了之前的模型元素，因此仍然可以识别原有的工作流过程定义，主要扩展的相关类包括 TimeEstimation、WorkTime、WorkTimeType、FixedTime、DynamicTime、DynamicTimeType、Normal、Possion、Exponet、Uniform 等，部分类的结构如图 5-4 所示。各类中包含了对添加的节点属性的一系列的 get 和 set 方法，用于获取和设置这些属性值。

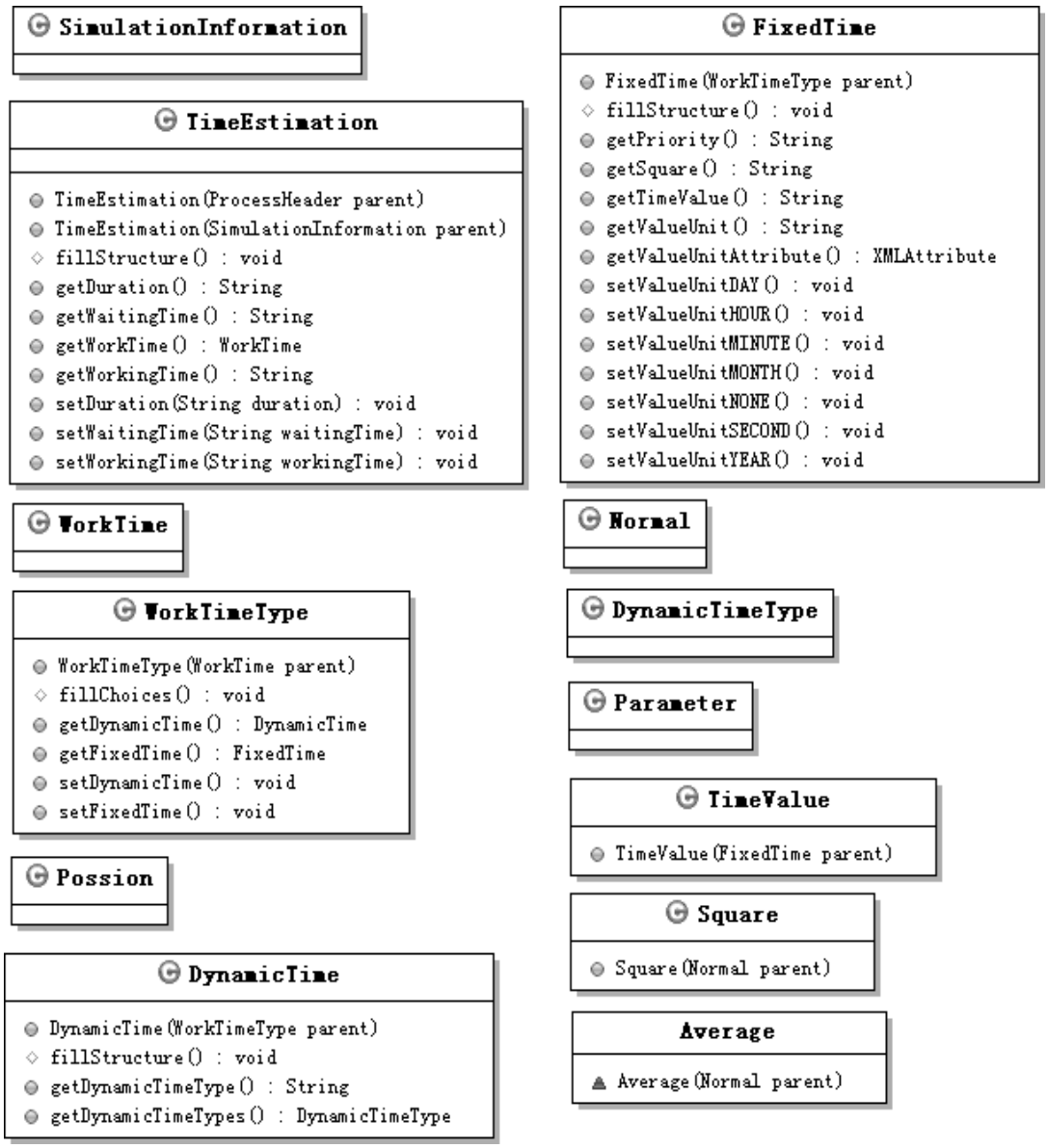

图 5-4 过程模型仿真元素的部分类图

在建立了相应类的基础上，通过提供的方法，在仿真模型的执行过程中活动的执行时间的解析过程的如图 5-5 所示。仿真引擎获得 simulationInformation 中的 WorkTime 对象，判断 WorkTime 对象的类型，如果为 FixedTime 则直接获取 FixedTime 的值并返回，如果为 DynamicTime 则需判断 DynamicTime 的类型，并根据不同的类型获取其设定的分布类型参数，最后调用随机变量生成器，随机变量生成器根据不同的分布类型及参数得到活动的执行时间并返回给仿真引擎。仿真引擎获得执行时间之后，自动完成该工作项，并设置活动时间为活动接受时间与该活动实例的执行时间之和，从而完成对活动的执行动作。

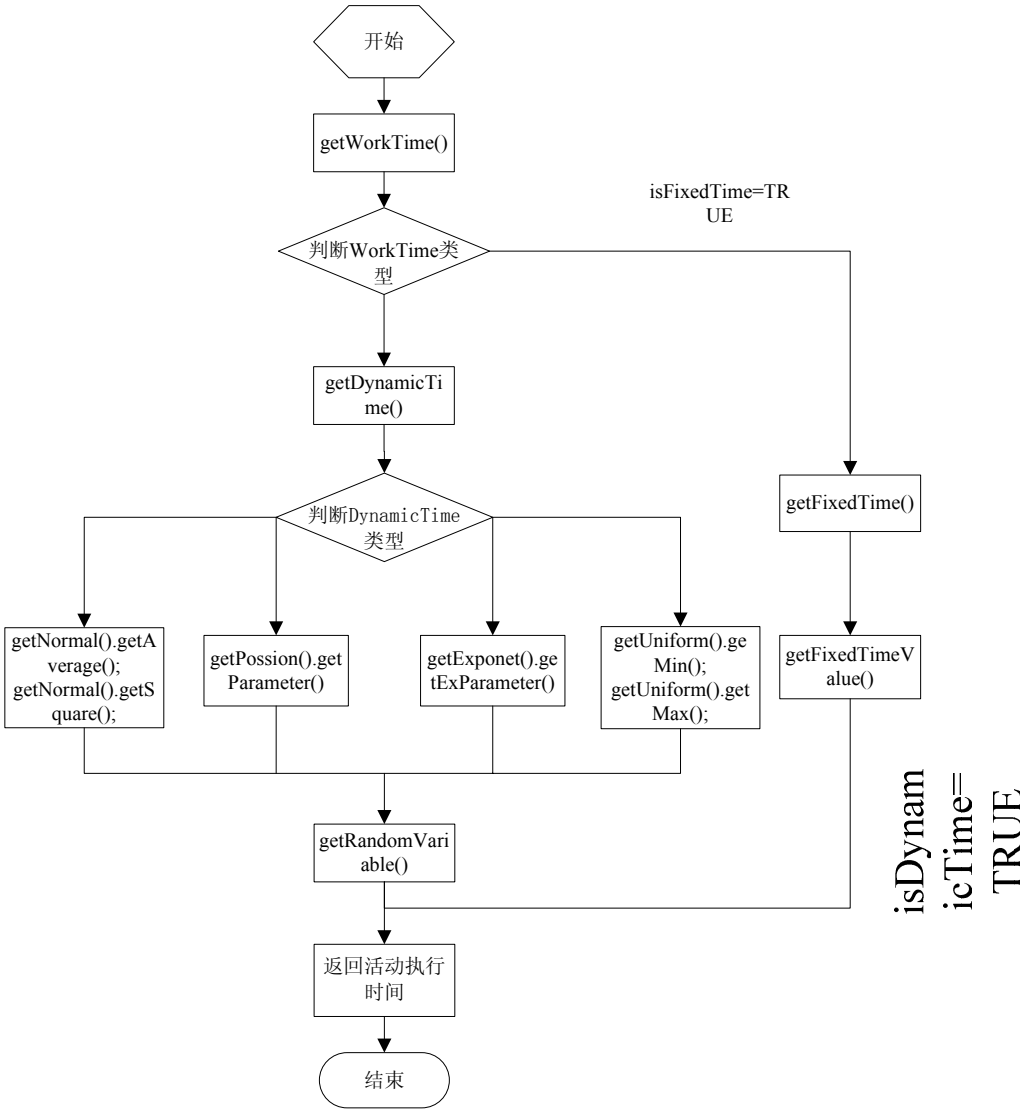

图 5-5 活动执行时间的解析

对选择分支的执行，如图 5-6 所示。活动 a_1 有 XOR 型输出分支，它含有 3 个输出分支，对于每个输出分支的选择是随机的。在具有概率型输出的活动执行之后，需要按活动的各输出分支概率选择一个分支来模拟实现现实情况。假定有 n 个分支，各分支 T_i 的概率为 f_i，显然有各分支的概率之合为 1，即 $\sum_{i=1}^{n} f_i = 1$。

isNormal=TRUE
isPossion=TRUE

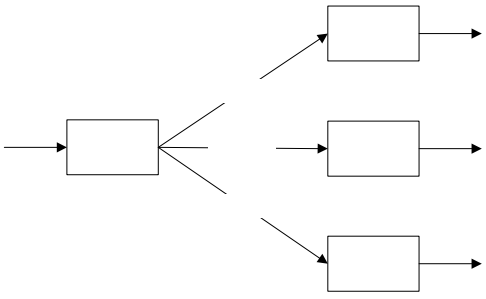

图 5-6 Split probability 示例

将区间(0,1)分成 n 个区间，其中第一个区间为$(0,f_1]$，第 j＋1 个区间为

$(\sum_{i=1}^{j} f_i, \sum_{i=1}^{j} f_i + f_{j+1}]$，其中 $2 \le j+1 \le n-1$。

从均匀分布随机数发生器中调用一个随机数，判断其落在哪一个区间，如果落在第 k 个区间，则第 k 个输出转移被选中。由于调用了均匀分布的随机数，因此当进行多次仿真时，可保证各引出任务被执行的概率符合所规定的数值。例如图 5-3，活动 a_1 有三组输出，其概率分别为 0.1、0.6 和 0.3，则选择执行活动 a_2 的概率范围为 (0,0.1]，活动 a_4 的概率范围为(0.1,0.4]，活动 a_3 的概率范围为(0.4,1.0)。若产生的随机数在(0,0.1]内则选择后续的执行活动 a_2，若产生的随机数在(0.1,0.4]内则选择后续执行的活动为 a_4，否则选择后续执行的活动为 a_3。

对于活动存在后续选择的情况，通过下面的操作可以获取 SIM-XPDL 中 Transition 属性，如图 5-7 所示。

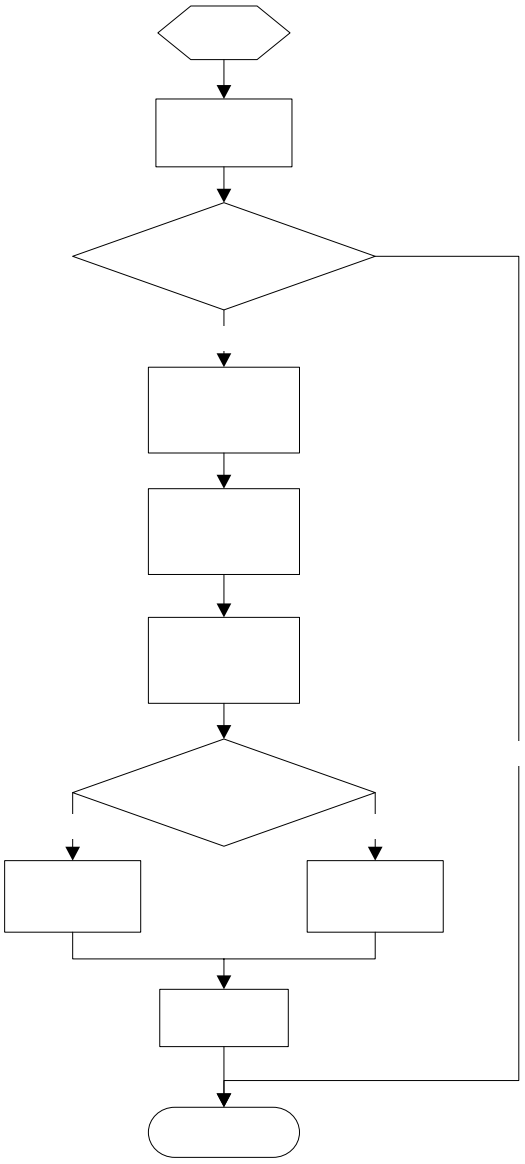

图 5-7 转移概率解析

5.2.3 随机数和随机变量生成器

在前面的章节中我们讨论了如何由观测数据确定随机变量的分布类型及其参数，仿真运行时，则需要根据确定的分布类型及其参数产生随机变量。产生随机变量的基础是产生(0,1)区间上均匀分布的随机数，也称为随机数生成器。其它各类分布，如正态分布、泊松分布等，都可以采用某种方法对这种均匀分布进行变换来实现。严格的说，仿真中采用的随机数生成器产生的随机数只能称为伪随机数。

目前使用的大多数随机数发生器是线性同余发生器，它是 Lehmer 在 1951 年提出的。线性同余法产生的伪随机数具有随机性好、易于计算机实现和速度快等特点而被广泛采用，详细的构建随机数发生器的算法可见文献[58]。

在获得随机数之后，对于不同的分布类型我们可以采用不同的产生随机变量的方法，如反变换法、组合法、卷积法和舍取法等[59]。其中反变换法是最常使用且最直观的方法，该方法的正确性证明可见相关离散事件仿真书籍[60]。在已知分布类型已经确定的情况下，从分布中取得采样值作为仿真模型的输入，其步骤如下：

1. 确定随机变量 x 的分布函数 $F(x)$；
2. 产生(0,1)区间上的均匀分布的随机变量 u；
3. 令 $u = F(x)$，解方程用 u 来表示 x；
4. 从而可以得到对随机变量 $u_1, u_2, ..., u_n$，由此获得的随机变量为 $x_i = F^{-1}(u_i)$。

5.2.4 历史数据管理器

历史数据管理器提供了对历史数据进行迁移、备份和分析的功能。其详细的设计在第四章进行了描述，结合历史数据特点对历史数据迁移和备份可以解决系统的性能问题而且为提取历史数据提供了便利。通过获取历史数据的接口可以提取我们所需要的数据，再结合统计知识对其进行分析获得其分析值，作为仿真参数的参考设置。

5.2.5 仿真时钟

仿真时钟用于表示仿真时间的变化，是控制仿真进程的时间机构。仿真时间的变化是基于仿真步长的确定，在仿真中推进的仿真步长可以是固定步长也可以是可变步长。在离散事件系统仿真中，其状态本来就只在离散时间点上发生变化，两个相邻发生的事件之间系统状态不会发生变化，因此仿真时钟可以跨过这些“不活动”的周期，从一个事件发生时刻推进到下一个事件发生时刻，仿真时钟

的推进出现跳跃性[50]。在这种方式下，通常在仿真开始时将仿真时钟设定为仿真开始时间或为零，随后仿真时钟按照一定的推进方式，按照下一个事件要发生的时刻不断给出仿真时钟的当前值，以不同的时间间隔向前推进，跳跃性地推进到下一个事件发生的时刻。本文中仿真时钟由时钟类 SimTime 来实现，提供了对时钟的操作方法。每当某一个事件发生时，系统开始处理相应的活动并计算出该事件触发产生的未来事件的发生时刻，经过一定活动处理时间后，仿真时钟将推进到下一个事件发生的时刻上，这个过程不断地重复，直到仿真运行满足规定的终止条件为止。

5.2.6 资源池

在本文中采用静态资源池，所谓静态是指我们在应用程序启动之初就建立了资源池。在仿真执行之前，根据企业经营过程中的组织模型和资源模型在资源池中根据角色类型建立其相应的资源。由于系统中存在着各种不同的资源，不同的资源有不同的特点，可以给资源添加附加的属性，如使用时间、资源被使用的次数和成本等。因此我们需要构建一个资源类 Resource，并需要一个高效的容器类来容纳系统中的资源对象。Java 中提供了如 Vector，Stack 和 Map 等容器类[61]可以方便地用来构建资源池。为了方便测试不同策略下的工作流模型的性能，可以定义资源管理策略。资源管理策略是指资源分配时，系统根据预定义的分配原则从可用的资源中取出最合适的资源。

5.2.7 仿真引擎管理器

仿真引擎管理器是整个仿真引擎中的关键组件，它负责整个仿真引擎各部件的调度和管理。其中仿真引擎管理器主要由类 SimManager 来实现，它实现了仿真执行的相关操作，包括初使化，启动，暂停和终止等，负责调度工作流引擎的接口以及执行仿真。

1. 总体仿真思想

工作流模型仿真属于离散事件系统的仿真，离散事件系统是指受事件驱动、系统状态跳跃性变化的动态系统，系统的状态变化发生在离散事件点上，这种系统往往是随机的。仿真引擎着重对人工型活动进行考虑，运行时，活动节点产生活动实例，不同的活动节点产生的活动实例在其生命周期的过程中会经历不同的状态，其中开始节点和结束节点以及虚活动节点产生的活动实例只有一种状态即完成（completed），人工型活动节点产生的活动实例包括三种状态：等待（waiting）、运行（running）和完成（completed）。

工作流模型仿真中主要存在的实体包括：过程实例、活动实例和用户。过程

实例在仿真中的状态包括运行（running）和完成（completed）；用户状态包括空闲（idle）和忙（busing）状态。运行时，活动实例产生后将处于 waiting 状态，当执行该活动实例的用户处于 idle 状态时，活动实例分配给用户，活动实例和用户的状态分别转换为 running 和 busing。当活动完成后，用户恢复为 idle 状态，活动实例转换为 completed 状态。在活动实例的生命周期中，活动实例需要经过等待用户、执行以及完成几个状态；工作流模型仿真包括的事件有：活动实例的产生、活动实例的完成和分配用户给活动实例。在这些事件中，用户属于互斥的资源，所以在同一时间内可能存在多个活动等待同一用户的执行，给活动实例分配用户具有不确定性。

为解决仿真过程中资源竞争的问题，我们利用资源池用于冲突消解。当存在多个并发的属于同一角色的活动时，如果该角色映射的资源可以满足当前的这些活动，则直接执行这些的活动。如果存在几个空闲资源，则根据调度规则选择需要执行的活动，其它活动等待这些资源的释放。由于在仿真过程中需要自动给出后续执行的活动，因此仿真引擎需要提供一定的调度规则。工作流仿真模型中对资源和组织模型都定义了其调度属性，本文采用的调度规则包括随机选择规则、FIFO 规则和优先级规则。其中随机选择是根据需要调用随机数发生器，产生随机数来确定下一个执行的活动实例； FIFO 是根据工作项生成时间选取最早被激活的活动实例作为下一步执行的活动。优先级规则，对于每个活动都定义了优先级，选取优先级最高的活动作为下一步执行的活动。

仿真引擎管理器执行仿真的基本过程如下：仿真开始时，开始节点执行，实例化开始节点后的活动，仿真时钟开始行进，并根据需要调随机变量生成器产生活动实例的执行时间，并作为下一个活动实例的开始时间，计算该活动实例的代价和成本；根据调度规则从该过程实例中的活动实例列表中选取下一步需要执行的活动实例；继续执行，直到仿真到结束节点，从而完成一次仿真。重复对上述过程进行 N 次仿真后，每次运行中各活动实例的随机抽样时间可能均不相同，过程模型执行的路径也会不一样，从而在每个活动节点上的统计数据都是一个随机样本，提供给仿真后的分析统计。本文在事务模型的基础上提供了对完成型触发机制的实现，其具体的仿真算法如图 5-8 所示。

2. 仿真算法

初始化设置：仿真执行次数为 k；根据工作流仿真模型建立的过程模型 P 会产生相应的过程实例 ID 表示为 P_{id}，而每个活动 a_i，$a_i \in P(i=1,2,...,n)$ 也会生成相应的活动实例 ID 表示为 A_{id}；设置仿真执行的最大次数 maxTrans；活动仿真信息中定义的活动执行时间为 WorkTime(i)；在执行过程中产生的活动实例 ai 的激活时间记为 ActivatedTime(i,A_{id})，活动实例接受时间为 AcceptedTime(i,A_{id})，

执行时间为 WorkTime(i,A_{id})，结束时间记为 LastStateTime(i,A_{id})，CurrentWorklist 存放了当前过程实例的活动实例列表。资源池 ResourcePool 存放了与业务过程相关角色映射的资源，包括资源所属角色、资源的使用时间及相关属性，其资源的使用结束时间（endtime）可以用来标识资源是处于空闲还是忙状态，每个资源的使用结束时间初始化为 0；设置仿真时钟开始时间 nowtime＝StartTime。

1. 令 k＝1；

2. 进行初始化设置，生成该模型的实例 P_{id}；

3. 取开始活动 a_0，记为当前执行活动；

4. 获取该活动的后续活动 a_1，生成活动实例 A_{id}，记 ActivatedTime(1,A_{id})=StartTime；

5. 判断该活动属于哪种类型，如果是虚活动，则设置活动的接受和结束时间为 AcceptedTime(1,A_{id})=LastStateTime(1,A_{id})=StartTime 并跳转到(7)，否则根据调度规则选取 CurrentWorklist 中的活动实例，并跳转到(6)；

6. 调用执行活动算法，执行该活动；

7. 调用 getNextTran()方法读取该活动的后继分支，如果为普通分支则跳转到(8)，如果为决策型分支则读取分支上的扩展属性 getFirstExtendedAttributeForName("SplitProbability").getVValue()，产生服从(0,1)均匀分布的随机数，判断该数落于哪个转移分支的区间，将随机数所在区间上的分支条件设为 true；

8. 将所有满足条件的输出分支指向的活动激活，不失一般性假定为活动 a_i；

9. 判断该活动是否为虚活动，如果为虚活动则设置活动的接受和结束时间为 AcceptedTime(i,A_{id})、LastStateTime(i,A_{id})为上一个调用它的活动的 LastStateTime 并跳转到(7)；如果为结束活动则跳转到(12)；否则记录激活时间 ActivitedTime(i,A_{id})，其中 ActivitedTime(i,A_{id})为上一个调用该活动的结束时间；

10. 获取当前过程实例下的所有激活状态的活动，根据调度规则选取 CurrentWorklist 中的活动实例；

11. 调用执行活动算法，执行该活动；

12. 判断该过程实例是否已经结束，如果该过程实例已经结束则跳转到(13)，否则跳转到(7)；

13. k=k+1；若 k 小于 MaxTrans，则跳转到(2)，否则转下一步；

14. 仿真结束。

图 5-8 完成型仿真算法

本算法没有考虑对仿真执行时间的约束，只定义了执行仿真的次数。对于仿真超时的情况可以通过线程不停的去检测仿真是否超时，如果达到仿真结束时间则抛出超时异常，并结束仿真。该算法属于串行离散事件仿真，考虑了具有同一角色类型的活动抢占资源的情况，完成型事务可统计经营过程的平均周期时间。其中对于活动是否可以执行的条件包括：该活动使用的资源是否空闲并且可用；该活动执行的条件是否满足。活动执行算法如图 5-9 所示。

1. 根据活动实例获取活动定义，判断该活动转入转移类型，如果为 AND-Join 转向(2)，如果为 XOR-Join 或无转入类型则活动转向(3)；

2. 获取该活动转入转移的最晚结束的 MAXLastStateTime，并设置该 LastStateTime 为该活动的激活时间 ActivitedTime(i,A_{id})=MAXLastStateTime；

3. 判断该活动实例是否可以执行，如果该活动实例的所需资源可用则记 AcceptedTime(i,A_{id})=ActivitedTime(i,A_{id})；否则获取资源的结束时间 endtime，设置该活动的 AcceptedTime(i,A_{id}) =endtime；

4. 获取仿真信息中的 WorkTime(i)，根据其设定的执行时间类型获得该活动的执行时间 WorkTime(i,A_{id})，调用随机变量生成器产生该工作项的工作时间，令 WorkTime(i,A_{id}) =randTime()，记录结束时间 LastStateTime(i,A_{id}) =AcceptedTime(i,A_{id})+WorkTime(i,A_{id})；并设置资源的使用时间 endtime=LastStateTime(i,A_{id})，如果 nowtime<LastStateTime，则推进仿真时钟到该活动结束时间，nowtime=LastStateTime。

图 5-9 *活动执行算法*

5.3 仿真实例

如图 5-10 所示，是某设计院的设计通用流程的过程定义，其中受理是一个转出转移为 AND-Split 型的活动（右边框为凸出的表示为 AND-Split），它并行触发财政局审批、设计院审批和工商局审批三个活动，然后这三个活动汇聚于处理活动（左边框为凹陷的表示为 AND-Join）。该路径活动判断三方审批是否通过，如果一致通过则进行统一算费，否则结束该过程实例。

工作流模型仿真过程包括业务过程建模、仿真环境参数设置、仿真执行。根据仿真执行步骤设置好仿真环境及其它必要参数、设定执行次数及仿真时间。对于普通活动及存在决策的路由需要设置相应的仿真参数，包括活动执行时间所属的类型，对于固定型则直接设置该活动执行时间，对于动态执行时间则设置该活动执行时间所属的分布类型及参数。

在本次仿真实例中，由于是一条新建的业务过程因此仿真参数由用户自行设定。其中受理服从参数为 0.5 小时的指数分布，工商局审计和财政局审计需要 7.3 小时的固定执行时间，设计院审计服从参数为 5.7 小时的指数分布，处理活动需要 0.5 小时，统一算费服从参数为 3.95 小时的指数分布，收费确认和办结分别需要 1 小时和 0.5 小时。在该过程定义中存在决策型的转移，其中设定处理指向统一算费的转移的概率为 0.7，指向虚活动的概率为 0.3。事务设置为完成型，调度规则采用随机型，建立过程模型的资源和组织机构，其中角色类型为设计院窗口人员的执行者包括 a 和 b 两个员工，工商局窗口人员和财政局窗口人员分别为 c 和 d，假定 a 员工处于工作时间表之内的劳动力成本为 10 元/小时，b 员工成本为 12 元/小时，c 为 5 元/小时，d 为 6 元/小时。定义员工的工作时间表为上午 8 点到下午 18 点之间。

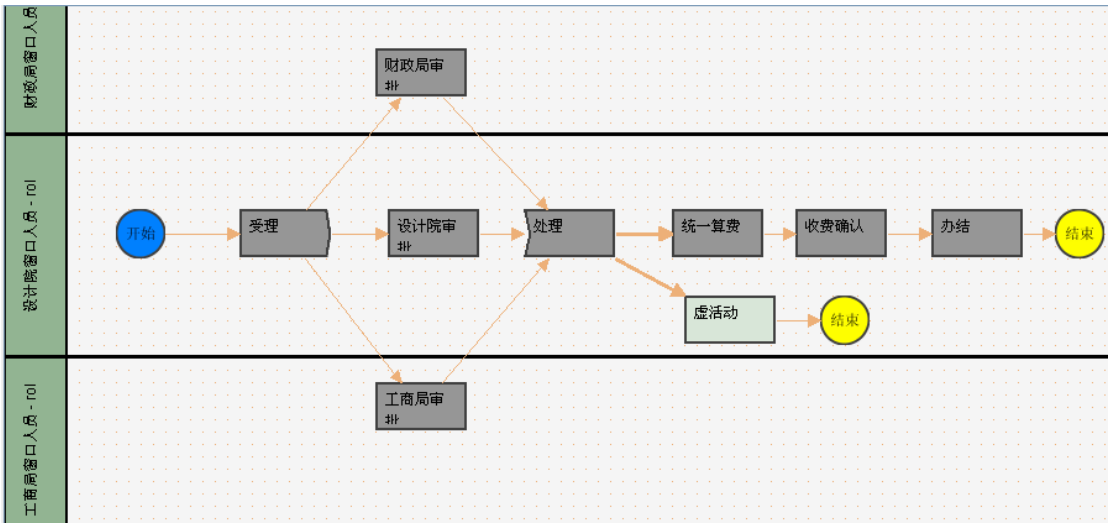

图 5-10 设计院通用流程在建模工具 JaWE 中的图形化表示

每一个活动的处理时间由工作时间和等待时间组成，工作时间指活动执行所需的时间，等待时间由等待资源时间和非活动时间组成，其中非活动时间指资源处于工作时间表之外的时间。

资源利用率是衡量资源使用状况的最常用的指标，我们忽略资源可能失效的情况下，资源利用率＝（资源工作时间）/（资源时间－非活动时间）。资源工作时间指资源实际使用的时间，资源时间在数值上指从仿真开始到仿真结束的这一段时间。

成本也是衡量企业经营过程的另一项重要的指标，通常采用基于活动的成本计算方法来统计经营过程的的总成本，如果根据资源类型来统计，成本由劳动力成本、设备成本、原材料成本和其它可计算的成本构成。

表 5-1 和表 5-2 是应用完成型仿真算法对过程定义进行仿真并按照上述的仿

真分析指标进行分析的结果，其中仿真次数为 N=10。在非活动时间内资源是不可用的，活动将暂停执行直到资源可用为止才继续执行。设定的仿真开始时间为2007 年 4 月 1 日 08 点 30 分，经过 10 次仿真后的结束时间为 2007 年 4 月 12 日 15 时 36 分，仿真总执行时间为 271.1 小时，非活动时间 154 小时。

表 5-1 活动执行时间分析

活动名称	执行次数	平均处理时间(小时)	平均等待时间(小时)	总成本(元)
受理	10	0.39	0.00	44.11
工商局审批	10	19.93	12.60	351.65
财政局审批	10	19.93	12.60	421.98
设计院审批	10	12.45	7.00	590.64
统一算费	8	6.66	3.50	285.02
收费确认	8	1.00	0.00	86.00
办结	8	0.50	0.00	90.00
处理	10	1.90	1.40	114.00
虚活动	2	—	—	—

表 5-2 资源利用率分析

角色	资源名称	利用率
设计院窗口人员	a	49.2%
	b	36.7%
工商局审批人员	c	62.5%
财政局审批人员	d	62.5%

在串行情况下，通过仿真分析可以看出设计院窗口人员的利用率不是很高。对于过程定义结构和功能的仿真，可以借鉴软件测试技术理论的思想。过程定义的功能可以表示为一组行为或动作的集合，可以仿照测试技术中的黑盒测试，对过程定义的执行过程中所需的输入参数进行等价类划分，仿真执行时依次读入所需参数，在该流程中，决策型转移的判断条件为当三方审批一致通过（Cs.pass==yes&&Ss.pass==yes&&Gs.pass==yes）才能执行收费确认，否则结束该流程。划分好等价类之后，在初使化过程实例之前事先设定好相关数据，仿真引擎执行时则是通过依次读入所需参数完成仿真，仿真结束后再判断流程是否完成了指定功能。对于过程定义结构的仿真，可以参考白盒测试，对执行过的活动和转移进行分析，一个工作流过程可以看成是一个由节点和连接弧所组成的有向图，因此对于结构的仿真可以分为活动节点覆盖仿真、分支覆盖仿真及路径覆盖仿真，具体的过程本文不做详述。

5.4 小结

基于开源项目 Shark、JaWE 设计了工作流模型仿真系统，该系统以工作流引擎和仿真引擎为核心。对系统关键组件进行了详细的介绍，并给出了完成型事务模型的仿真算法。该系统可以支持 SIM-XPDL 描述的工作流仿真模型，并给出了串行仿真的实例，对仿真结果进行了分析。

第六章 总结与展望

6.1 本文总结

自上世纪九十年代初业务过程重组的概念提出后，过程工程被公认为是企业提高生产率、改善产品质量、增强竞争力的关键。因此为了在竞争中持续保持优势地位，企业十分注意改进和优化业务过程。无论采用哪种思想理论对企业业务过程进行优化构造，都必须以对业务过程有充分的认识。工作流模型仿真是对业务过程建立模型并动态仿真内部各种行为活动的一种方法，对企业进行过程革新、改进乃至过程集成提供有有力的决策支持。本文以工作流技术为支撑、涉及到支持仿真的工作流模型、过程定义语言、仿真参数设置、仿真模型解析、验证及执行等方面内容。具体研究工作及创新点总结如下：

1. 通过对传统工作流理论的扩展，提出了支持仿真的工作流过程定义元模型，并在前人工作的基础上提出了结合工作流历史数据分析的支持仿真的工作流模型，从而为工作流模型仿真的进一步研究提供了理论基础。

2. 讨论了目前工作流的建模语言，在深入研究工作流管理联盟元模型和过程定义语言 XPDL 标准的基础上，通过对 XPDL 提供的仿真元素进行了兼容性扩展，增强了 XPDL 对仿真执行过程中不确定性信息的描述能力。

3. 对工作流执行历史数据进行分析，分析了工作流历史数据的特点，对工作流历史数据进行迁移和备份解决了系统性能问题。并在此基础上提出了利用工作流历史数据的分析结果作为工作流仿真环境提供参考设置，可以为工作流模型的仿真方便地建立其仿真环境。

4. 提出了一个较为通用的仿真引擎，建立了工作流模型仿真系统。基于离散事件系统仿真的仿真机制，通过研究离散事件系统仿真原理，构建了支持 SIM-XPDL 的工作流仿真引擎。仿真运行过程中随机变量生成器、仿真时钟的推进以及仿真算法的研究为仿真系统的实现奠定了基础。通过对 WfMC 的 XPDL 的扩展以及对开放源代码的工作流管理系统的研究，以工作流引擎 Shark 为基础，完成了支持工作流模型仿真的模型建模、验证、仿真环境设置、执行的工作流模型仿真系统。

6.2 未来工作

随着建模与仿真理论和方法的研究不断深入，以及作为支撑技术之一的计算机技术的不断发展和进步，对仿真技术有了新的要求，仿真软件的发展目标一直

是不断提高面向问题、面向用户的模型描述能力及改善它对模型建立、实验、分析、设计和检验的功能，从而使得仿真方法有了一些新的发展。就本文来说，还主要是停留在重复级上、仿真些相对简单的业务流程，如果要进一步提高工作流仿真的能力，还需要不断的研究：

1. 仿真模型验证是提高仿真可信度的基础，仿真的可信度不仅取决于模型本身，还取决于输入参数的可信度，本文对工作流执行历史数据进行了分析从而得到仿真环境的参数设置，但是如何对工作流历史数据的进行进一步挖掘，如何在此基础上进行灵敏度分析，从而校准仿真模型，这些都有待于进一步研究的问题。

2. 仿真分析的性能指标包含很多类型，对于如何合理全面的建立性能指标体来帮助用户进行仿真分析也是需要进一步的考虑。

3. 目前无论是仿真研究，还是关于工作流的研究均从集中式转向分布式，这为引入 Agent 完成工作流仿真提供了可能。各个 Agent 接受任务后，可以某一时间长度、以特定的策略和质量水平完成任务，因而呈现出分布式仿真的特点，这将使过程仿真在深度和广度上得到扩展。

4. 本文对资源和组织以及其调度规则等方面的因素加以了考虑，但考虑的情况还比较简单，有些功能没有完全实现（如仿真数据的统计分析显示等），因此未来的研究工作需要进一步完善仿真系统的功能。

参考文献

[1] 夏火松，蔡淑琴. 基于信息资源管理的企业流程再造. 情报搜索，2001，(4)：47~49

[2] 范玉顺. 工作流管理技术基础. 北京：清华大学出版社. 2001

[3] 邹宇.工作流模型仿真及验证技术的研究：[硕士学位论文].江苏：南京航空航天大学，2003

[4] 王寿欣，覃正，张磊. 基于工作流的企业过程分析与重组方法研究. 西安交通大学学报，2001，35（4）：430~434

[5] WfMC. WfMC-TC-1003. Workflow Management Coalition Workflow Standard: The Workflow Reference Model. Winchester, UK: Workflow Management Colition, 1995

[6] 李红臣，史美林. 工作流模型及其形式化描述. 计算机学报，2003，26（11）：1456~1463

[7] Bruce Gladwin, Kerim Tumay. Modeling Business processed with simulation tools. In: Proceedings of the1994 Winter Simulation Conference, 1994. 114~121

[8] 施勇. JTang 工作流服务仿真技术研究：[硕士学位论文]. 浙江：浙江大学，2006

[9] Tarumi H, Kida K. Ishiguro Y, et al. WorkWeb System Mutil-Workflow Management with a Multi-Agent System. In: Proceedings of ACM International Conference on Supporting Group Work, 1997. 299~308

[10] Joon-Soo Bae, Seok-Chan Jeong, Youngho Seo, et al. Integration of Workflow Management and Simulation.Computers and Industrial Engineering, 1999, 37: 203~206

[11] Matinea-Garcia A I. A Process Model Simulation Tool-Mapping from Process Models to Discrete Event Simulation. In: Proceeding of the 12th European Simulation Multiconference, 1998. 849~853

[12] 胡宝月. 基于功能网的工作流管理系统模型建立器和仿真器的设计与实现：[硕士学位论文]. 辽宁：东北大学，1997

[13] 范玉顺，马权，张军. 基于集控 Petri 网方法的面向对象建模与仿真工具. 清华大学学报（自然大学学报），1998，10（38）：89~92

[14] 刘轶铭，范玉顺. 基于工作流的企业过程的建模和仿真技术研究. 清华大学学报，2000，1（40）：89~92

[15] LIN Huiping, FAN Yushun,WU Cheng. Research of Agent-Based Workflow Simulation. In: AMSMA International Conference, Guangzhou, 2000. 753~756

[16] 钟琦. 层次型工作流仿真模型的研究与实现：[硕士毕业论文]. 湖南：国防科学技术大学，2004

[17] 傅谦. 支持软件过程改进的工作流仿真研究：[博士学位论文]. 上海：上海交通大学，2002

[18] Frank Leymann, DieterRoller. Business Process Management with Flowmark. IBM Corporation,1996

[19] SIMPROCESS. Process Modeling and Analysis with SIMPROCESS. http: //www.caciasl.com/docs/UsersManualHTML43/Output/SPUser-04-1.html

[20] F.Casati, P.Grefen, B.PERNICI, et al. WIDE Workflow Model and Architecture. 1997. http://www.sema.es/projects/WIDE/Documents/

[21] David harel, Eran Gery. Executable Object Modeling with Statecharts. IEEE Computer, 1997, 30(7): 31~42

[22] Stef Joosten. Trigger Modeling for Workflow Analysis. In: Proceedings of the ninth Austrain-information conference on Workflow management, 1995. 236~247

[23] Y. Lei, P. Singh. A Comparison of Workflow Metamodels. 1998. http://osm7. cs.byu.edu/ER97/workshop4/ls.html

[24] K.Meyer-Wegener, M.Bohm. Conceptual Workflow Schema. In: Proceeding of IFCIS International Conference on Cooperative Information System, 1999. 234~242

[25] 吉海峰，范玉顺. 面向仿真的工作流模型. 航空技术制造，2004，（1）：49~51

[26] 孙滔. 面向仿真的工作流模型及关键技术的研究：[硕士学位论文]. 浙江：浙江大学，2005

[27] WFMC. Interface 1: Process Definition Interchange Process Model(WFMC TC-1016-P), WFMC, 1999

[28] WFMC. Interface 2: Workflow Client Application Application Programming Interface (Interface 2 & 3) Specification(WFMC TC-1019).

[29] WFMC. Interface 4: Interoperability (Wf-XML 2.0).

[30] WFMC. Interface 5: Audit Data Specification (WFMC-TC-1015).

[31] 邹冰，张旭. 一类基于 IPO 的工作流过程建模语言规范分析. 计算机工程与科学，2002，38（21）：54~58

[32] 朱瑜，杨国伟. 过程定义语言 XPDL，BPML 和 BPEL4WS 的比较分析. 成

都信息工程学院学报，2005，20（6）：645~648

[33] 徐庆，袁兆山，潘秋菱. 基于 XML 的工作流过程定义语言模型 XMWPDL. 小型微型计算机系统，2003，24（5）：848~852

[34] Das S, Koehut K, Miller J, Sheth A, Worah D. ORBWork: A reliable distributed CORBA-based workflow enactment system for METEOR2. http://lsdis.cs.uga.edu/

[35] DavidHelmbold, David Luckham. TSL: task sequencing language. In: Proceedings of the 1985 annual ACM SIGAda international conference on Ada, 1985, (2): 255~274

[36] Rajesh k,Thiagarajan,etc. BPML: A Process Modeling Language for Dynamic Business Models. In: Proceedings of the Fourth IEEE International Workshop on Advanced Issues of E-Commerce and Web-Based Information Systems, 2002. 222~224

[37] Francisco Curera, Rnaia Khalaf, et al. Implementing BPEL4WS: the architecture of a BPEL4WS implementation. Concurrency and Computatio: Practice and Experience, 2006, 18(10): 1219~1228

[38] WFMC. XPDL 2.0: Intergrating Process Interchange and BPMN. http://www.WfMC.org/ researchreports/articleswhitepapers.htm

[39] 马华. 分布式应用集成中的面向服务工作流研究：[硕士学位论文]. 湖南：中南大学，2006

[40] 刁洪滨. Lotus Workflow 工作流机制初探. 哈尔滨师范大学自然科学学报，2005，21（2）：54~56

[41] 工作流建模工具 CIMFLOW-Designer 介绍. http://www. simflow.net/software/CIMflowtool/CIMFlowDesigner.htm

[42] 赵文，胡文蕙等. 工作流元模型的研究和应用. 软件学报，2003，14（6）：1052~1059

[43] Ehydra JaWE. Open Source Java XPDL Editor. http://www.enhydra.org/workflow/jawe/index.html

[44] SourceForge. Net: Nezha. http://sourceforge.net/projects/nezha/

[45] 许甸，李建华等. 支持多实例模式的动态工作流研究. 计算机应用，2006，（12）：3033~3036

[46] Compatibility form FOLCOC. http://foldoc.org/index.cgi?query=compatibility&action=Search

[47] WfMC. WfMC-TC-1025.Workflow Management Coalition Workflow Standard:

Workflow Process Definition Interface--XML Process Definition Language (XPDL 2.0). Florida, USA: Workflow Management Coalition, Lighthouse Point, 2005

[48] 郭禾，孟令剑，陈锋. 工作流仿真平台的设计与实现. 计算机工程，2001，27（6）：104~105

[49] 康凤举，杨惠珍，高立鹅等.现代仿真技术与应用. 北京：国防工业出版社，2006：114~126

[50] 肖田元，张燕云，陈加栋等. 系统仿真导论. 北京：清华大学出版社，2001：161~215

[51] Lai Jin, Fan Yushun. Workflow Logs Analysis System for Enterprise performance measurment. High Technology Letters. 2005, 11(3): 274~279

[52] Wil van der Aalst, Ton Weijters, Laura Maruster. Workflow Mining: Discovering Process Models from Event Logs. In: IEEE Transactions on Knowledge and Data Engineering, 2004, 16(9): 1128~1142

[53] Onoda S, Yumoto M, Maruta T, et al. Bottleneck Detection analysis for workflow improvement. In: IEEE Transactional Conference on Computational Cybernetics and Simulation, 1997, (4): 3331~3336

[54] Eder J, Olivotto G E, Gruber W. A Data Warehouse for Workflow logs. In: Proceedings of the First International Conference on Engineering and Deployment of Cooperative Information Systems, 2002.1~15

[55] 王冬青，欧阳昱，刘玉树. 基于历史信息的工作流过程改进方法. 计算机仿真，2005，22（4）：46~48

[56] ObjectWeb. Enhydra Shark. http://shark.objectweb.org/

[57] 林慧苹，范玉顺，吴澄. 支持企业经营过程重组的工作流仿真技术研究. 信息与控制，2001，30（1）：11~15

[58] 何光渝，高永利.Java 常用数值算法集. 北京：科学出版社. 2003

[59] 刘伟军. 基于工作流技术的业务流程的仿真评估. 计算机工程与应用，2003，35：79~80

[60] Jerry Banks, John Carson. Simulation of Discrete Event System. Beijing: Tsinghua University Press. 1988

[61] 飞思宝兰研究院. JBuilder 精髓. 北京：电子工业出版社. 2004

致谢

在我的课题和硕士论文完成之际，谨向在攻读硕士学位过程中曾经指导过我的老师，关心过我的朋友，关怀过的领导，和所有帮助过我的人们致以崇高的敬意和深深的感谢。

首先，要感谢我的导师李建华教授，感谢他三年来在学习、生活和科研工作中给予我的关怀、指导和照顾。李老师治学严谨、积极工作、以身作则、以及不断学习新知识的精神使我在学业和为人上受益匪浅。同时，感谢李老师给予我参与项目实践的机会，使我的动手能力有了较大的提高，这将对我以后的工作产生积极的影响。李老师对本论文的选题和研究提出了许多有益的建议，使我能够顺利完成论文。

感谢高琰师姐、马征师姐、马华师兄和徐海军师兄，他们在我的项目实践、论文研究及撰稿期间提出了很多宝贵的意见。感谢郝丽波、许甸、曹龄兮、丁昭华、杨萍、何毅俊和李金同学，感谢师弟师妹的支持和帮助，跟他们在一起学习、工作、讨论将是我终身怀恋的事情。另外，感谢项目组的同事陶格等，为本论文的系统实现给予了指导和帮助。

同时，感谢我的家人为我的成长所付出的心血，以及多年对我的教育和培养；感谢其它亲人和朋友对我的鼓励和关心，感谢帮助过我的所有人们！

攻读学位期间主要的研究成果

已发表的学术论文：

[1] 夏媛，李建华. 基于历史数据分析的工作流仿真环境研究.计算机仿真.拟刊在2008年3月

[2] 夏媛，李建华. 工作流性能的分析方法.计算机应用研究.拟刊在2007年7月

[3] 许甸，李建华，刘星沙，夏媛.支持多实例模式的动态工作流研究.计算机应用，2006，（12）：3033~3036

曾参加的科研项目：

[1] 参与了南县政务审批系统的设计与开发

[2] 参与了常德卷烟厂企业信息门户系统的设计与开发

[3] 参与了开源工作流引擎 Shark 的研究和改进工作