基于 Web Services 工作流管理系统新模型①

Workflow Management System of New Model Based on Web Services

马文涛 陈 虹 (辽宁工程技术大学 电子与信息工程学院 辽宁 葫芦岛 125105)

摘 要： 在充分分析传统工作流管理联盟参考模型等相关标准的基础上，讨论了工作流的基本概念。针对目前工作流管理系统交互性弱，无法解决分布异构环境中数据传递与表示的问题，以工作流管理系统参考模型为依据，提出了一个基于 Web Services 的工作流管理系统新模型，该模型采用面向服务的分层架构分解复杂的业务流程，提高了系统的描述能力和跨平台能力，能够适应信息管理系统业务过程的不断重组问题，提高了应用系统开发效率。

关键词： 工作流 Web Services 面向服务 跨平台 流程

1 序言

工作流技术是工作流管理系统中的核心技术，也是提高业务过程效率和生产率的关键技术。工作流技术的提出缓解了传统管理系统业务流程臃肿的问题，它所具有的协调技术决定了其在复杂业务的信息过程中将发挥重要的作用。采用工作流技术作为核心开发的业务管理系统可以按照企业的具体需求快速生成应用软件系统，并且在客户业务过程中根据需要进行业务流程重组。这只是优化了传统企业管理系统难以解决的复杂化问题，但现代企业管理信息系统不再是只面向企业内部功能，更多的是面向市场，面向客户。目前市场上的工作流管理系统模型大多数描述能力和跨平台能力有限，很难适应软件系统移植性和扩展性的特点。针对这种情况，本文提出了一个基于 Web Services 的工作流管理系统新模型，更大程度上挖掘了工作流技术的潜力，提高了软件系统的描述能力和跨平台能力。

2 工作流管理系统参考模型

工作流起源于生产组织和办公自动化领域，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，其目的是通过将一个具体的工作分解成多个任务、角色，按照一定的规则和过程来约束这些任务的执行和监控，以提高企业管理系统水平。根据工作流联盟(WFMC)的定义，工作流是一类能够完全或者自动执行的管理过程，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.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，并按照预先定义好的工作逻辑完成工作流实例的执行。工作流技术是流程建模和流程管理的核心，为了实现工作流技术的标准化和开放性，使得不同的工作流管理系统之间能够进行信息交换和协作，WFMC 提出了一个共性的工作流管理系统参考模型。如图 1 所示。

WFMC 提出的工作流管理系统模型是一个具有广泛意义的参考模型，针对不同的企业管理信息系统,软件开发商都会在此基础上对其进行改动。目前许多工作流管理系统模型是为企业应用而应用，很少考虑软件性能的优化与软件功能的移植性，因此在模型描述上都采用自己专门的工具对其进行开发，从软件工程角度来说就是对模型的描述能力过低。另外，许多工作流管理系统参考模型对复杂业务流程的建模能力及

① 收稿时间:2008-12-08

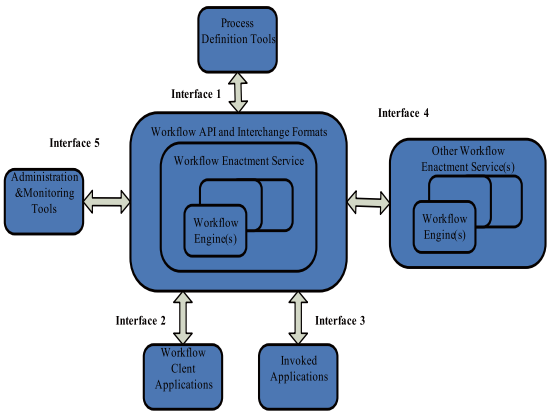

图 1　工作流管理系统参考模型

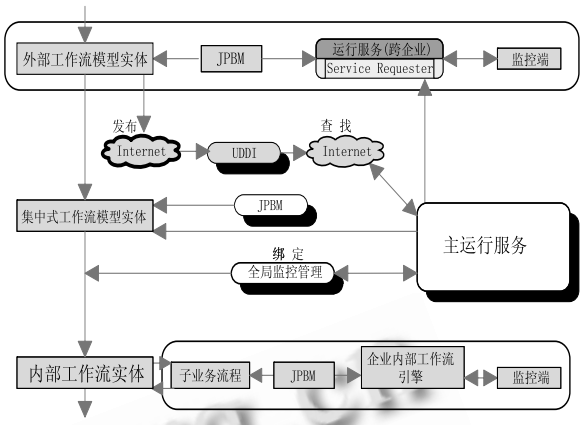

图 2　基于 Web Services 的工作流总体架构图

业务流程之间的交互能力有限，很难满足跨平台的分布操作和复杂业务信息化的要求，使得软件产品的重用性不高。为了更好地实现工作流管理系统的目标领域，采用目前软件领域较为成熟的 Web Services 技术，建立一种新的工作流管理系统模型,并使用 Java、JBPM 等服务技术对其进行实现，以激发工作流技术的优越性。

3　工作流管理系统新模型

在工作流管理系统参考模型的基础上，建立一个基于 Web Services 的工作流管理系统新模型，这种体系同传统的工作流模型相比，有两大优势：①适应性广。Web Services 利用标准的 Internet 协议和 XML，XML 的内容与表示的分离使得处理起来灵活方便。②真正的平台，语言独立性。因为 Web Services 的接口就可以进行服务的请求与调用。基于这些特点，以下讨论实现一种基于 Web Services 的工作流管理系统新模型。

3.1 基于 Web Services 的工作流管理系统体系结构

传统的集中式工作流模型包括活动(Activity)，转移条件(Transition Condition)，角色(Role)，工作流相关数据(Relevant Data)等基本实体。本文提出的基于 Web Services 的工作流模型，相对与传统的集中式工作流，对相关模型部分实体进行了进一步的抽象。其体系结构如图 2 所示。

(1) 集中式工作流模型实体：即传统的集中式工作流模型的活动实体。

(2) 外部工作流模型实体：即调用远程 Web 服务的实体,用 Web Services 技术来实现。

(3) 内部工作流实体：相当于调用企业内部其他工作流引擎上的相关过程的实体。

本设计方案在保留原有集中式工作流数据的基础上，增加了与调用 Web 服务和调用其它工作流引擎相关的数据，整个系统的体系结构图如图 2 所示外部工作流模型实体所调用的子流程是由跨互联网的企业伙伴的运行服务建模，解析和执行的，被封装喂 Web 服务，供主流程调用完成一个步骤。集中式工作流模型实体相当于传统的工作流模型实体。内部工作流实体用来调用同一企业内部其他工作流系统上的相关流程。

3.2 工作流建模

工作流的建模方法是否得当决定了在任意情况下定义工作流流程是否清晰，是否满足用户在建模过程中所提出的各种要求。业务流程中子流程的建模是分布在其他工作流系统上的。不同的企业伙伴各自完成其内部流程模型的建立,并只对外公开这些内部模型的相关交互接口，以便实现不同企业工作流模型之间的连接。这样的工作流模型是自底向上综合和自顶向下分解两种方法的组合。自顶向下的分解过程完成整个跨工作流系统、跨企业流程的分解，各个企业伙伴完成其内部流程建模后，采用自底向上综合进行模型的连接和匹配，从而形成整个工作流模型。主业务流程建模时，对于外部过程节点，若对 Web 服务采用静态调用，在建模时直接指定其 URI，若按动态绑定方式，即在流程运行时，向注册中心设定一个查询条件，依据查询结果通过 SOAP 协议和 Web 服务中的远程对象绑定,实现调用,在建模时需指定查询条件;对于内部过程节点，设置相关调用参数，并将返回结果设置到

流程模型中。本文提出一种基于 Web Services 的过程元模型，如图 3 所示。

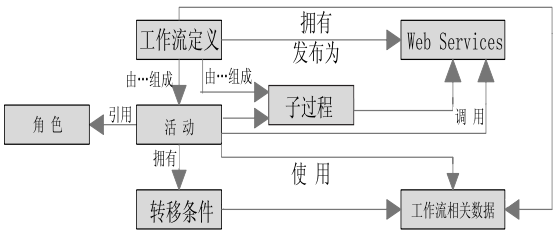

图 3 基于 Web Services 的过程元模型

这种模型相对于传统的工作流元模型，变化主要体现在以下几点：

(1) 对活动概念进行了推广，相对传统集中式工作流中的活动，因为活动可以调用被发布为 Web Services 的子过程,所以这里的活动隐含包括子过程。

(2) 传统的集中式工作流中的相关数据都是在同一个运行服务上，活动之间不需要数据映射，而在分布式异构环境下，各工作流系统所使用的数据格式可能不统一，而要在各活动之间进行数据的传递和操作，有时需要建立相应的映射关系。

(3) 将跨组织的工作流发布为 Web Services，而 Web Services 又可以被工作流所调用，实现了流程的合并和递归调用。

3.3 工作流引擎

工作流引擎负责解析流程定义并且控制流程实例的运行。当流程实例满足一定条件后被启动，引擎调用相应的应用程序，在关联活动间传递数据信息和控制信息，并在活动执行的过程中生成工作项，从而推动整个业务流程的运行。各个工作流引擎之间以及工作流引擎与用户 Web 界面之间采用 SOAP 消息的方式进行交互。图 4 给出了基于 Web Services 的工作流引擎体系结构。

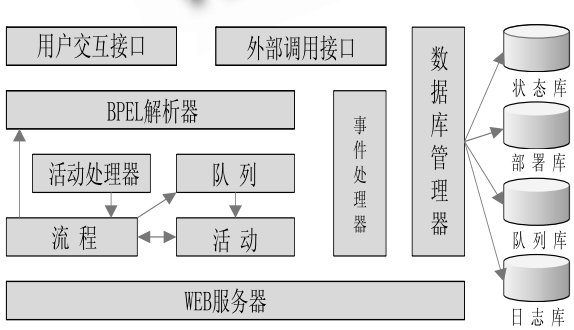

图 4 基于 Web Services 的工作流引擎的体系结构

从图 4 可以看出，工作流引擎主要由 BPEL 解析器、活动处理器、用户交互接口、服务调用接口、数据库管理器等构成。

(1) BPEL 解析器负责从流程定义文档、Web Services 定义文档、其他资源文档读取流程定义的所有信息，将其转化为内部格式后存入数据库，并负责装载成流程定义对象，使业务流程处于就绪状态。

(2) 流程活动处理器负责与活动相关的处理。包括活动状态的转移控制、工作项的生成和删除等。流程活动处理器主要有三个部分：流程、队列和活动。工作流引擎运行一个或多个 BPEL 的流程，这些流程由活动组成，而这些活动也可以包括活动。队列负责储存暂时信息。流程活动处理器根据接收到的消息执行流程，并同步或异步地产生响应消息或其他调用消息。它基本上实现 3 个任务：接收消息、更新内部状态、发送消息。

(3) 用户交互接口提供的操作包括完成对任务表和任务项的处理。包括按照特定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、查询一个任务项的属性等。完成对流程实例状态的查询，例如获得某个流程或活动实例的详细信息。查询和改变工作流流程、实例的运行状态、修改流程定义和工作流流程定义。

(4) 事件管理器主要监听流程执行过程中的执行事件，记录流程实例或活动实例的执行历史，分别记录全局事件、流程实例执行事件、各种活动实例执行事件。并将事件持久化到关系数据库。还负责从数据库中读取事件日志数据，用于流程实例监控。

(5) 服务调用接口的流程中会涉及对各个伙伴的交互，这些交互就是对交互方的 Web Services 或者是业务流程子流程的调用。同时，由于流程和伙伴是对等的，交互方也可能反过来调用(回调)流程，这些调用操作都是通过服务调用这个接口进行的。对于这个接口来说，涉及的调用都为 Web Services 调用操作，这样就简化了接口的设计。

(6) 数据库管理器要包括四个数据库，状态库用来存取流程的状态，在流程重新启动过程中，将主要访问这个数据库。队列库用来存取消息和流程的排队情况。部署库用来存取部署的文件信息，部署的状态等。日志库用来存取日志信息。工作流引擎采用了插入式的结构，不同的管理器可以执行不同的存储机制。

4 模型的实现与分析

基于 Web Services 的分布式工作流对实体进行了更高层次的抽象，它的执行也是分布的。总体业务流程的引擎获得整体过程定义后，主引擎根据过程定义中的相关信息，将整个业务流程分解为一个个的节点配置信息，并把配置信息发布到与各个节点对应的工作流服务器。对一般活动节点的解释、执行与集中式工作流相同，外部过程节点和内部过程节点所调用子流程的执行是分布在不同的工作流运行服务系统上的，每个运行服务系统完成其对应的子流程执行。

4.1 跨企业子流程的分布执行

Web Services 是基于服务的，当主工作流引擎作为服务请求者要调用相关服务时，它首先到 UDDI 注册中心查询到满足要求的 Web 服务后，到其地址，并将相应的 WSDL 文件下载到本地工作流服务器上，当主工作流引擎需要相应服务时，就根据获得的地址发起连接，然后通过 SOAP 协议和 Web 服务中的远程对象绑定在一起，进行请求的发送和应答的接收。发送和接收的都是符合 SOAP 规范的 SOAP 消息。在 SOAP 的 Headers 和 Body 中即可传送工作流相关数据，实现外部工作流的调用。

4.2 企业内部子流程的分布执行调度算法

当工作流主引擎要调用企业内部的子工作流流程时，可以通过分布式构件层对子流程进行直接的调用。通过主调用服务器的事件触发器调用远程子过程事件，发出调用请求；当被调用的服务器接受到消息后，如果通过了消息解析器的解析，则会创建并发送相应的过程实例；主调用服务器如果收到了创建成功的实例消息，则会向被调用服务器发送设置相关数据请求消息；被调用服务器会根据收到的请求信息进行相关的操作；主调用服务器获得过程实例启动成功的消息后。会根据主调用方式的不同作不同处理。同步交互采用先完成后调用的方式，异步调用采用先调用后执行的方式，并行同步交互采用先集合后执行的方式。

4.3 基于 Web Services 的工作流可调度性分析

时间特性分析是分析工作流过程定义的重要组成部分。工作流可调度性分析的目的是判定工作流过程定义是否满足时间约束，其时间行为是否正确。可调度性分析即可在过程定义时进行，以判断对活动指派的时间约束是否合理，也可在过程实例的执行过程中进行，以支持系统对工作流活动的调度。

工作流可调度分析的主要内容有两个方面：其一是对流程中的活动在系统给定状态下是否可调度进行判别；其二是分析任一活动对整个流程的时间约束的影响，判断是否违反了时间约束。

相对触发时间域反映系统一个状态改变为另一个后续状态所有可能的时间约束，以此为依据可以判断个别活动的可调度性。为了分析活动的调度对整个流程时间约束的影响，利用绝对触发时间域，以反映系统从初始状态改变为当前状态的后继状态所有可能的时间约束。状态 M_i 下变迁的绝对实施域记为：AD_i $(1\leq i\leq n)$。变迁 t 在 AD_i 中的时间区间记为 $AD_i(t)=(AEFT_i(t),ALFT_i(t))$或 $t(AEFT_i(t), ALFT_i(t))$，其中 $AEFT_i(t)$和 $ALFT_i$ (t)分别称为 t 的绝对最早触发时间和绝对最迟触发时间。要计算绝对触发时间域，需引入全局时间戳。在一个变迁触发后，可达标识被打上时间戳(变迁的绝对触发时间区间就是系统在该变迁触发后所到达的标识的时间戳)。假定 $TS_i=(AE_i,AL_i)$是 M_i 的时间戳，表示相对于 M0 的标识时刻 θ，M_i 在时间区间 $TS_i=(AE_i,AL_i)$内可达。令初始状态 M_0 的时间戳为 $TS_0=(0,0)$, $AD_0(t)=\{C(t);t\in EN(M_0)\}$。

为了分析流程活动一个可能的调度分配 S，由时间约束工作流网模型相应的无时间网中得到与 S 相对应的一个从开始状态到结束状态的状态变迁序列 $\sigma=M_0,t_1,M_1,t_2,\ldots\ldots,t_n,M_{n+1}$,根据工作流网的定义，此变迁序列一定是存在的。通过检查状态变迁序列 $\sigma=M_0,t_1,M_1,t_2,\ldots\ldots,t_n,M_{n+1}$,中每一个变迁 it 的时间约束特性，可对流程活动的调度 S 的可调度性进行分析。

将时间参数引入工作流模型中并对模型进行时间维上分析是工作流技术的重要研究内容。我们提出了基于 Web Services 的工作流过程可调度性分析的方法，将可调度性分析与活动执行时间的估计结合起来，为工作流的高效执行提供了理论和方法上的支持。这里提出的方法不仅可用于建模时的时间约束验证，也可用于工作流执行时的任务调度分派和流程的时间控制。因此，在系统的建模工具中加入验证功能，对已有模型进行验证，在工作流定义中加入活动时间信息以及全局信息。工作流建模完成后，对模型进行静态分析。在工作流引擎中可以采取最小时间触发，进一步验证了基于 Web Services 的工作流管理系统的高效性。

5 集成的WEB服务安全模型

由于 Web Services 是一个新兴的技术，所以在安全方面并没有完整的规范，而在工作流管理系统的实际应用中，也许要涉及到多个 Web Services 的集成，本文在分析了现有安全技术的不足和 Web Services 的安全需求后，设计了一个集成的 WEB 服务安全模型。

5.1 安全模型的体系结构

通过将现有的安全规范与技术集成到一个框架中，实现安全目标。同时模型具有一定的开放性，当应用环境与安全技术发生变化时，能够通过模型自身的调整自动适应所发生的变化，体系结构如图 5 所示。

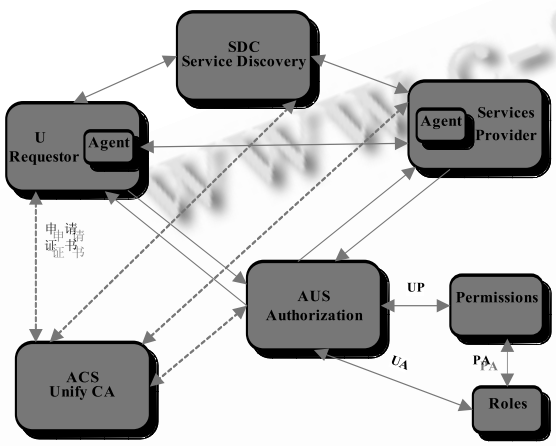

图 5 基于 Web Services 的安全模型的体系结构

5.2 安全模型的定义

集成的 WEB 服务由六元组{U,R,P,S,ACS,AUS}组成，其中的对象描述如下:

(1) U 表示所有服务请求者的集合；

(2) R 表示所有角色的集合；

(3) P 表示所有权限的集合，指 Web Services 可以进行的操作；

(4) S 表示一系列可访问 WEB 服务的集合；

(5) ACS 表示认证服务；

(6) AUS 表示授权服务；

5.3 安全模型的实现过程

根据上述定义，安全的 Web Services 调用的过程分为用户注册并申请证书、发布服务、发现服务并申请授权、服务访问等 4 个步骤。为了保证系统的安全性，发出服务请求的用户 U 与服务提供者 S 都要向 ACS 申请各自的证书及公匙/私匙对，除了服务发现中心以外，所有的交互双方都要进行双向身份认证。当调用远程 Web 服务时，需要先注册并申请证书，在获取证书后，向服务中心申请发布服务信息，服务请求者便会获得 URL，然后有 AUS 为其分配角色从而获取授权；当调用企业内部工作流引擎时，只需执行企业的内部验证即可。

6 结束语

Web Services 是一种基于面向服务的新理念，在许多标准和安全方面并没有统一的规范，但由于它的平台独立性和语言独立性，随着人们的不断探索和研究，Web Services 必将在分布式工作流的研究和实现中发挥越来越重要的作用，并将逐渐成为主流。

参考文献

1 梁爱虎.精通 SOA:基于服务总线的整合应用开发.北京:电子工业出版社, 2008.

2 褚红伟,葛玮.基于 Web Services 的分布式工作流的研究与实现.计算机应用与研究, 2004,(8):49－51.

3 Delap S. CloudCamp's Reuven Cohen Discusses Virtualization and Cloud Computing. 2008－7. http://www.infoq.com/news/2008/07/cohen .

4 jBPM-jPDL 学习笔记—流程设计与控制. http://liangyi2007.javaeye.com/blog/176345.2008－03－26.

5 Little M. InfoQ Interviews BPEL4People Representatives.2008-4.http://www.infoq.com/articles/bpel4people－tc.

6 Lai R.周斌,等译. J2EE 平台 Web Services－Java 技术丛书.北京:电子工业出版社, 2005.