翻译人：张敬波 （网名：踏冰） OICQ：42508298

Email：say4ever2u@yahoo.com.cn MSN：tabingfly@hotmail.com

翻译人：韩 伟 （网名：浆糊） OICQ：3413384

Email：java_cn@21cn.com MSN：Java_cn@21cn.com

欢迎访问我们的网站： http://www.EasyWorkflow.org

工 作 流 管 理 联 盟 规 范

WFMC ——工作流参考模型

文档编号 TC00-1003

95 年 1 月 19 日

目 录

1. 简介

1.1. 背景

工作流管理系统一项快速发展的技术，各种行业渐渐的采用工作流技术。工作流技术的主要特点是：过程的自动化处理，这些过程包含由人与以机器为基础的活动相结合；特别是对那些与 IT 应用程序、工具交互的过程，进行自动化处理。虽然，工作流技术广泛用于办公环境中，例如保险、银行、法院和行政管理等，然而工作流技术，也可以应用于一些类型的工业和制造业。

许多软件开发商都有 WFM 产品，并且不断有新的 WFM 产品走入市场。市场上可选择的产品范围很大，因此每个开发商只关注产品特殊功能，而用户可以采用不同的商品来满足不同的需求。然而，没有统一的规范使得不同的 WFM 产品协同工作，这是由于不兼容的过程控制方式所导致。

WFMC 是由一些公司联合到一起成立的组织，从事上述问题的研究。业界一直认为，所有的 WFM 产品都有一些相同的特性，这样对各种功能使用公共的标准，就可以实现不同产品间的协同工作。WFMC 的成立是为了确定那些功能范围，并且为 WFM 产品的实现制定适当的规范。希望，这些规范能够使不同种类的 WFM 产品协同工作，并提高工作流应用程序与其他 IT 服务（例如，Email、文档管理等）的集成能力，从而工作流技术在 IT 市场中被更有效的使用，开发商与用户共同受益。

1.2. 目的

本篇文档的目的是：提出支持上述文档开发的框架。为“工作流管理系统（workflow management systems）”提出了一个公共“参考模型（Reference Model）”，来确定特性、术语和组成部分，并且在工作流系统的完整模型范围内，可以对开发单个组成部分的规范。详细的规范将作为单独的文档来开发。

1.3. 范围

本文档包含工作流管理系统的感念、术语、通用结构、主要功能组件、接口以及在接口间进行交换的信息。本文档指出了，工作流管理系统适用标准化的部分，并且描述了通过采用公共的标准可以实现系统间协同工作的情况。本文也讨论了，现有标准对工作流管理系统的适用性，与其他标准 IT 服务的集成性。但本文不包含更广业务过程机制。

1.4. 对象

本文是面向 WFMC 成员的，还有那些对工 WFMC 努力感兴趣并想理解支撑 WFMC 工作的顶层技术结构的人。有适当技术的人可以参阅本文，但并不需要对工作流系统有一定的知识。

1.5. 如何阅读

第 2 章对工作流系统的感念做了概括的介绍，接着介绍了相关的商业，以及可以集成工作流技术的系统的背景。如果你不熟悉工作流技术，应该从第 2 章开始阅读。如果熟悉工作流管理系统，可以从第 3 章开始。

第 3 章讨论工作流系统的内部结构、主要功能组件、和他们的交互性。介绍了上层体系结构，以及各种接口，这些接口用来支持不同系统组件间的协同工作，用来支持与其他主要 IT 基础组件的集成。

第 4 章概括介绍了工作流应用编程接口（WAPI—Workflow Application Programme Interface），并介绍了支持开放互操作的必要协议，讨论与规范一致的原理。

1.6. 参考

WFMC SC00 - 1002 WFM Coalition Proposal Information

WFMC SC00 - 1006 WFM Coalition Technical Committee Operations

WFMC TC00 - 1008 Interoperability White Paper

WFMC TC00 - 1009 Client application API descriptions

WFMC TC00 - 1010 Workflow Definition Read/Write Descriptions

WFMC TC00 - 1011 Terminology and Glossary

WFMC TC00 - 1013 Workflow APIs - Naming Conventions

1.7. 修订历史

2. 工作流系统简介

2.1. 什么是工作流（workflow）

“工作流”干预过程、业务程序的自动化处理，文档、信息或者任务按照定义好的规则在参与者间传递，来完成整个业务目标或者对整个业务目标的完成做贡献。同时，“工作流”可能由手工组织，实际上，多数“工作流 ”都在IT系统中进行组织的，从而对过程自动化提供计算机支持，WFMC把工作定位在这个方向上。

“工作流”定义：

全部或者部分，由计算机支持或自动处理的业务过程。

工作流经常与“过程重组（BPR—Business Process Re-engineering）”联系在一起。BPR 是关于企业（组织）核心业务过程的评估、分析、模拟、定义以及其后的操作实现。尽管，不是所有的BPR都是采用工作流实现的，但工作流技术是最佳的方法，主要因为，工作流技术提供了业务过程逻辑与IT操作支持的分离，从而以后可以修改过程规则来重定义业务过程。相反，工作流技术并不只在BPR中采用，例如用于现有的业务过程中。

“工作流管理系统（WFMS—Workflow Management System）”通过管理工作活动序列，调用与各种活动步骤相关的人员、IT资源，对业务过程提供自动化处理。

“工作流管理系统”定义：

工作流管理系统是这样的一个系统，详细定义、管理并执行“workflows”，系统通过运行一些软件来执行workflows，这些软件的执行顺序由工作流逻辑的计算机表示形式（计算机化的业务规则——过程定义）驱动。

每个业务过程都有一个生命周期，从几分钟到几天（甚至数月），由过程的复杂性与组成活动的持续时间来决定。有多种方法实现工作流管理系统，使用多种IT和通讯组件，运行环境可以从一个小的本地工作组到企业间。因此，WFMC参考模型从各种角度考虑工作流管理系统，希望提供各种不同的实现技术、运行环境。

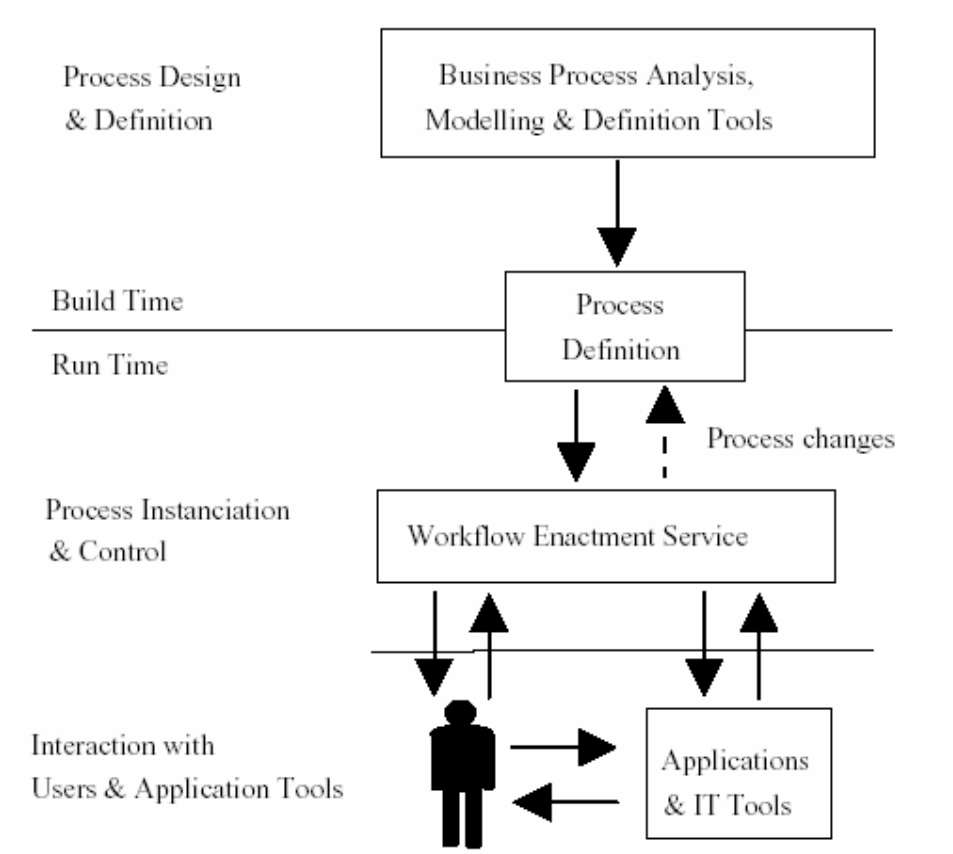
尽管实现的方法多种多样，但所有的WFMS都表现出某种共同的特性，这为不同产品间的集成、协同工作提供了基础。参考模型描述了工作流系统的一个公共模型，并且指出参考模型如何能使其与其他各种不同的实现方法相关联。

在最高层，所有的WFMS都相同的特性，即为下边的3个功能提供支持：

- 建立时期（Build-time）功能，定义、模拟工作流过程，及其组成活动。

- 运行时期（**Run-time**）控制功能，在运行环境中管理工作流过程，管理不同活动执行顺序。
- 运行时期与用户、**IT**应用程序（工具）的交互，来处理各种活动的执行。

下图描述**WFMS**的基本特性，以及上述功能间的关系：



2.1.1. 建立时期功能

建立时期的功能负责产生业务过程的计算机化定义。在这个阶段，通过使用一个或多个分析、建模和系统定义工具，把实际中的业务过程转变成形式的、计算机可以处理的定义。通常把定义的结果称为过程模型、过程模板、过程元数据、或者过程定义。在本文中，称为“过程定义（process definition）”。

“过程定义”定义：

过程的计算机化表示，包括手工定义和工作流定义。

过程定义由一些列的分散的活动、相关的计算机、人员操作、活动间控制过程进程的规则构成。可以用文本、图形或者语言符号来表示过程定义。有些工作流系统允许在运行时期改变过程定义，在上图中如反向箭头所示。

WFMC没有把过程定义的初始阶段作为规范的一部分。但这是区分不同WFM产品的重要标志。然而，过程定义被看作是规范的一部分，以实现不同的建立时期工具（过程定义工具等）与运行时期的产品间交换过程定义数据。

2.1.2. 运行时期过程控制功能

在运行时期，过程定义由负责创建、控制过程实例的软件所解释，这个软件并负责安排过程中各个活动的执行时间，调用适当的人员、IT应用程序资源等。这些运行时期的控制功能，就象过程定义中描述的过程与现实所见到的实际过程间的联接，反映在运行时期的用户与IT应用程序间的交互。核心组件是基本工作流管理控制软件（工作流机—engine），负责过程的创建与删除，控制运行过程中活动的执行时间安排，以及与人、应用工具资源进行交互。工作流机经常是分布与多个计算机平台中的，用来处理在大的地域跨度中操作的过程。

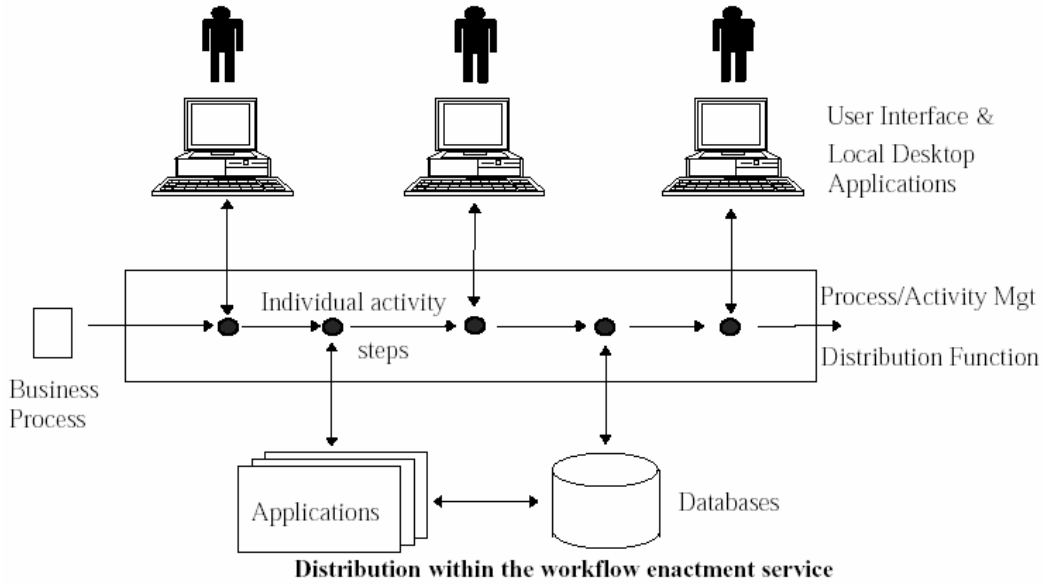
2.1.3. 运行时期活动交互

典型的，工作流过程中的活动与人类的操作有关，交互经常是在使用特殊的IT工具后，或者信息处理操作需要一个特殊的应用程序来操作一些定义过的信息时，交互才被意识到。在活动间转移控制、确定过程的运行状态、调用应用工具、传递适当的数据等，都必须与过程控制软件进行交互。制定标准的框架来支持这种交互，有如下就个好处：在多工作流系统中使用一致的接口；可以开发工作于不同工作流产品中的通用应用工具。

2.1.4. 分配与系统接口

在参与者间分配任务和信息的能力是工作流运行时期组件的主要区分特性。分配功能可能在各种不同的级别上进行（从工作组到企业间），这要依靠工作流的范围；也许会使用多种不同的消息机制（电子邮件、消息传递、分布式对象技术等）。另一种强调这种分配问题的上层工作流体系结构如下图：

下图中，工作流制定服务（任务分配）是核心功能组件，通过接口提供给用户、以及分布在工作流范围内的应用程序。每一个这样的接口都是一个潜在的、可以与其他工作流制定服务、其他基础组件或应用程序组件进行集成的点。



Distribution within the workflow enactment service

在工作的流程中也许包括，在不同的开发商的工作流产品间传递任务，以使业务过程的不同部分能在不同的平台或者使用特殊产品满足特殊过程阶段的子阶段中运行。在这种情形下，中间方框中的流程会在两个或者多个工作流产品间传递。例如，活动1，2和5可能会在一个工作流系统中执行，而活动3，4由其他工作流系统执行，同时控制在适当的点，在不同的工作流系统间进行传递。支持这种工作流控制传递的标准，使得可以利用几个不同的工作流产品协同运行来开发联合式的工作流应用。

WFMC定义的所有接口如下：

- 过程定义数据，以及过程定义数据的转换规范
- 支持不同工作流系统间协同工作的接口
- 支持与各种不同IT应用程序交互的接口
- 支持与用户交互的接口
- 提供系统监视，以及标准功能来简化复合工作流应用环境管理的接口

2.2. 工作流的发展

2.3. 产品实现模型

简介

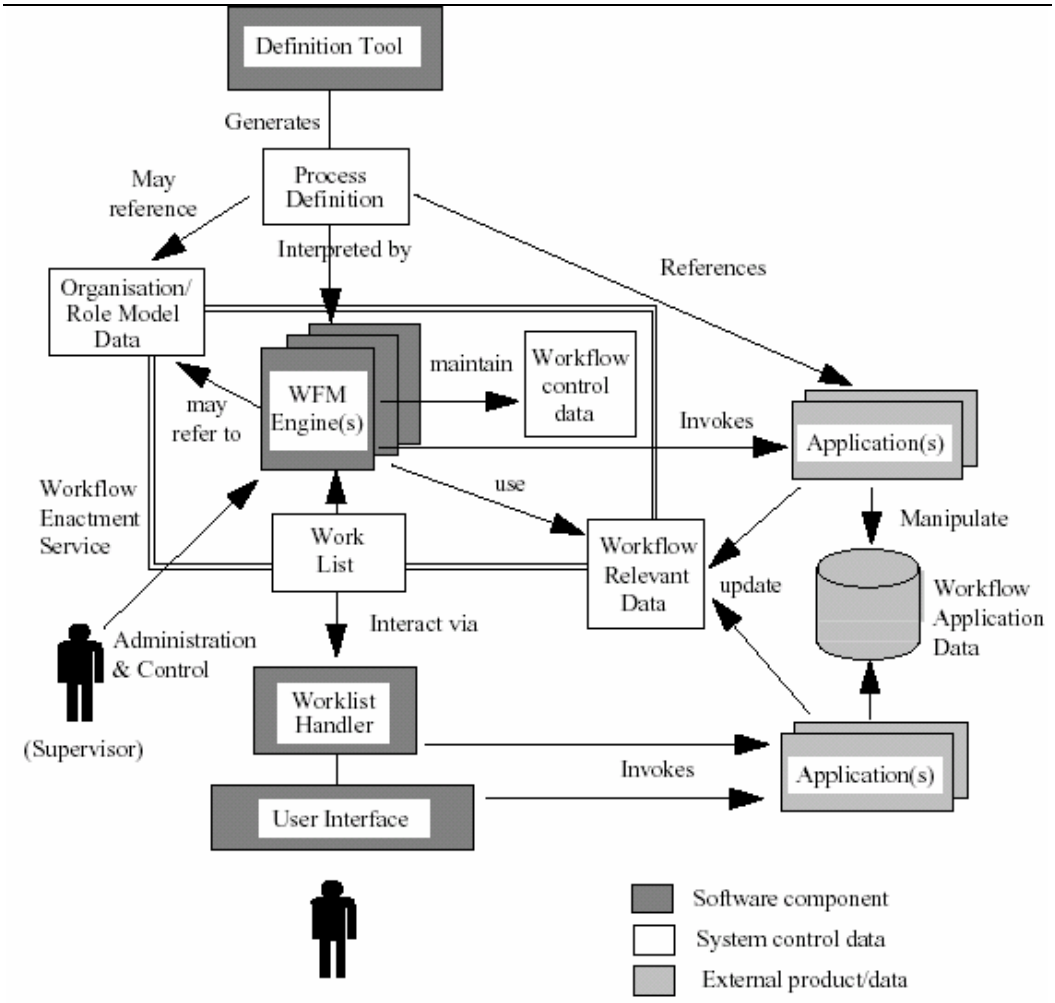
尽管市场上的工作流产品是各种各样的，但是已证明可以构建一个通用的工作流系统实现模型，这个模型可以适用于市场上的大多数产品，因此为开发协同工作的工作流系统奠定了基础。

把工作流系统中的主要功能组件，以及这些组件间的接口看成抽象的模型。考虑到会有许多其他的具体实现不同于这个抽象模型，因此，特殊接口在不同的平台中所采用，还有分配技术。而且并不是所有的开发商都会暴漏功能组件间的每一个接口；这会由不同一致级别的规范来处理，规范会详细定义互操作功能，并有开放接口来支持多开发商产品的集成。

通用工作流系统的主要功能组件，如下图：

通用模型有3种类型的组件：

- 软件组件，为工作流系统的各种功能提供支持
- 各种类型的系统定义和控制数据，被一个或多个软件组件所使用
- 应用程序和应用程序数据库，其并不是工作流产品的一部分，但是他们会被工作流产品调用，从而作为整个工作流系统的一部分



Generic Workflow Product Structure

过程定义工具（Process Definition Tool）

过程定义是用来创建一个计算机可以处理的形式的过程描述。可能要以形式过程定义语言、对象关系模型、简单的系统、脚本、或者在参与者间进行信息传递的路径集为基础。工作流定义工具，可能作为工作流产品的一部分、也可能作为业务过程分析产品的一部分来提供给用户，作为业务过程分析产品一部分，会有其他的组件来负责处理业务过程的分析或者模型，这时，必须要有兼容的转换格式，与运行时期的工作流软件进行过程定义的相互转换。

过程定义（Process Definition）

过程定义包含，工作流执行软件运行过程所需的过程所有详细信息。包括过程的开始和结束条件、组成活动、在活动间进行导航的规则、需执行的用户任务、可能会被调用的应用程序、所有工作流相关数据的定义等。

过程定义可能会涉及到一个组织/角色模型，模型包含组织结构和组织中的角色等信息。从而使过程定义在，与具体活动或信息对象相关的组织实体和角色功能方面，十分详细。工作流执行服务器负责把工作流运行环境中的参与者与相应的组织实体或角色联系起来。

工作流执行服务器（Workflow Enactment Service）

工作流执行服务器软件负责，解释过程定义、控制过程实例、安排活动的执行顺序、向用户工作表中添加工作项目、调用应用工具。这需要一个或者多个协同工作的工作流机来完成这些职责，工作流机管理各种过程的一个单独实例。工作流执行服务器维护内部控制数据，这些数据或者集中于一个工作流机中，或者分布在一个工作机集合中；这些工作流控制数据包括与各种过程、或者正执行的活动实例相关的内部状态信息，也包括工作流机用来合作或者从失败中进行恢复的检查点、恢复/重新启动信息。

过程定义与（运行时期）工作流相关数据协作，一同用来控制过程中活动的导航、提供活动的进入与退出条件、不同活动的并行执行、顺序执行选项、用户任务、与每个活动相关的IT应用程序等。如果过程定义包括组织模型/角色实体类型，那么完成以上任务，需要访问组织/角色模型数据。

工作流机也包括调用一些形式的应用工具的能力，来激活必要的应用程序执行相关活动。这种调用机制间有很大的不同，在一些简单的系统中，也许只提供对单一的固定工具调用（例如，文本编辑器），然而在工作流系统中可能提供调用本地与远程的大范围内工具的方法。

工作流相关数据和应用数据（Workflow Relevant Data and Application Data）

过程导航判断或工作流机中的其他控制操作，都以工作流应用程序产生或者更新的数据为基础，这些数据可以被工作流机和条件工作流相关数据（也成为情况数据）所访问；这是工作流机唯一可访问的应用程序数据。尽管，工作流机负责在应用程序间传递工作流应用程序数据，但工作流应用程序数据直接由被调用过程操作。不同的应用程序由工作流过程内的不同活动调用。

任务表（Worklists）

过程执行中需要用户交互的地方，工作流机把任务添加到任务表中，以便任务表处理器对其处理，任务表处理器管理与工作流参与者的交互。这个过程对工作流参与者可能是不可见的，任务表在工作流软件中维护，把用户需要执行的下一个任务提供给他。在其他系统中，任务表可能对用户是可见，用户自己从任务表中选择执行任务，任务表也用来指示任务的完成。

任务表处理器 用户接口（Worklist Handler & User Interface）

任务表处理器是一个软件组件，管理工作流参与者与工作流执行服务器间的交互。任务表处理器负责请求用户关心进展中的任务，并负责通过任务表与工作流执行服务器进行交互。在一些系统中，只是使用一个桌面应用程序来提供一个简单的任务进入，等待用户注意。在其他一些系统中，任务表的处理可能更成熟，控制任务在一些用户间进行分配，并考虑到转载平衡、任务重分配等。另外的一些任务表处理功能，工作流机典型支持与客户端应用程序大范围的交互，包括工作流参与者的签到和退出、请求过程实例的开始、任务排队等候特殊的参与者，等。在工作流参考模型中，更广泛的使用“客户端应用程序”这个词，而不是“任务表处理器”，从而反映其潜在的广大使用范围，其包含任务表处理功能的同时也包含过程控制功能。

在图中，用户接口是一个单独的软件组件，负责提示和处理用户对话框，并控制本地用户的本地接口。在某些系统中，用户接口可能会与任务表处理器组合到一起，构成一个简单的功能实体。我们希望一些客户端应用程序能够和几个不同的工作流服务器进行交互，从而把服务器中的任务整理成统一的格式，通过公共用户接口提供用户。

可能会必须调用本地应用程序，来支持用户完成特殊的任务，这由任务表处理器来负责，或者由用户负责，在用户接口使用简易通用工具来安装适当的支持程序。在任务表处理器/用户接口中调用应用程序与工作流执行软件直接调用应用程序，有明显的不同。

管理操作（Supervisory Operations）

工作流系统中有许多的管理功能；这些管理功能以工作站点或者用户的管理权限为基础。这些管理功能使得管理者可以修改任务分配规则、确定过程中组织角色的参与者、跟踪遗漏的最终期限报警或根据其他事件、跟踪某一过程实例的运行历史、查询任务吞吐量或其他统计信息，等。使用分布式工作流机的地方，可能需要特殊的命令来在不同的工作流机间传递控制操作或者（局部）响应，从而提供一个单一的管理接口。

外部和内部接口（Exposed and Embeded Interfaces）

上述的体系结构适用于大多数工作流产品，但是并不是所有的产品在每个不同的系统功能组件间，都提供外部接口；一些产品把几个功能组件作为一个逻辑实体来实现了，并把接口包含在了软件组件的内部，导致无法被第三方产品使用。WFMC规范定义了每个接口在实现多工作流系统协同工作中的作用，因此，可以鉴别单独的产品是否符合协同工作标准。

2.4. 可选择的实现方式（Alternative Implementation Scenarios）

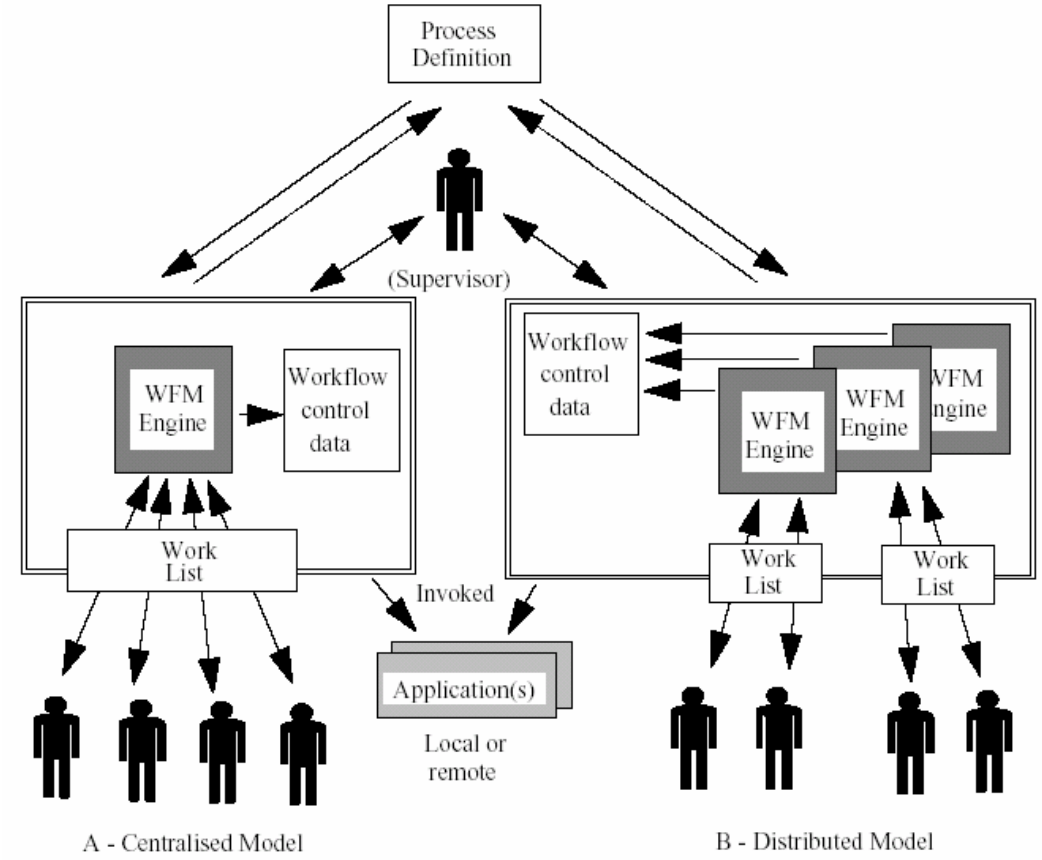
通用的工作流产品实现模型，定义了一系列的软件组件和接口。在一个具体产品实现中，可以采多种不同的方法来实现通用模型；这是产品间的一个重要区别点。产品间的主要的区别因素有：平台、网络基础构件的选择、工作流软件自身的继承功能。本部分讲述，通用处理通用模型的不同实现方法，同时保留可见接口来方便多开发商产品间的协同工作。

对所有的可能实现方案进行讨论，已经超出了本文的范围。主要考虑的可选择实现方法的部分如下：

- 集中式或者分布式的工作流执行服务器
- 任务表处理器的位置与分配机制

工作流执行软件—可选择的实现方法

工作流执行软件由一个或者多个工作流机组成，这些工作流机负责管理全部（或部分）过程实例的执行。可以用一个工作流机建立一个集中式的系统来管理所有过程的执行，或者建立一个分布式的系统，多个工作流机协同工作，每个工作流机管理所有过程执行的一部分。



Standard workflow enactment service boundary

在上图中，两个工作流服务器在边界处都表现了公共的属性，但是却采用了不同的内部实现结构。

在过程实例的运行中几个工作流机合作的地方，与过程实例相关的控制数据必须能够被不同的工作流机所访问。这些工作流控制数据可能会分布在多个工作流机中、或放置在一个主工作流机中、或作为共享资源来存储。同样在过程执行时，过程定义数据也可能分布在多个工作流机中，或者从主存储源中部分地传递到每个独立的工作流机中。处理管理操作或者应用程序调用的接口，可能分布式或本地化（集中式）地提供给工作流机。这种管理工作流在多工作流机中分布的实现方法是很多的，并且是十分复杂的。

WFMC采用的方法是在工作流执行服务器周围定义一个边界，边界表现出多种不同的标准功能属性，并通过一套标准的**API**可以对其访问。**WFMC**没有定义执行服务实现这种功能的内部机制，执行服务可能会包含一个或多个同质的工作流机，并使用多种通信方式。

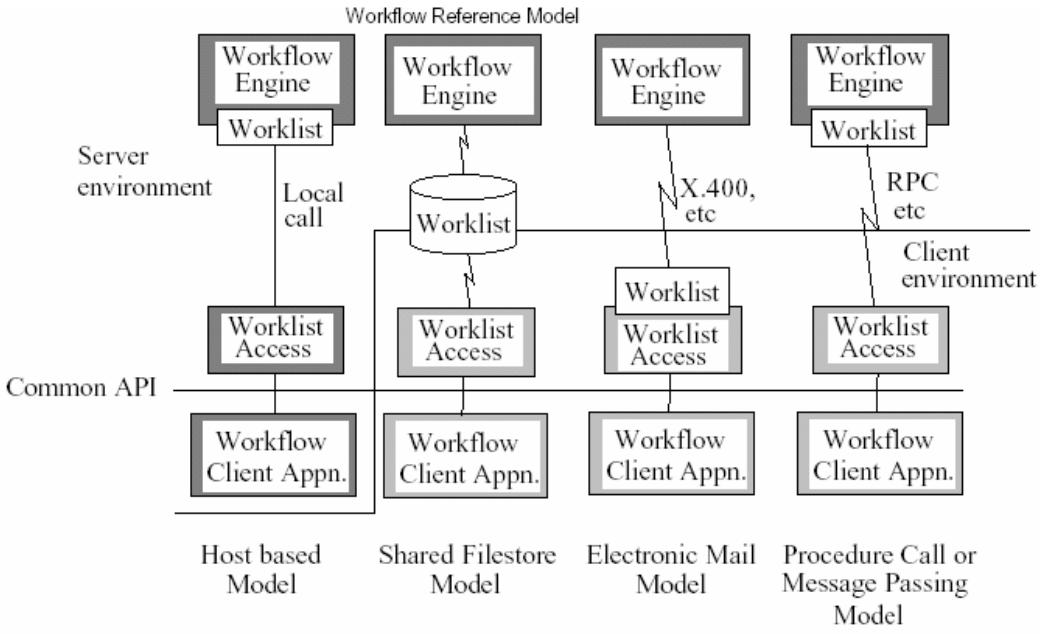
为了支持不同产品间的互操作， 定义了一些用来实现不同执行服务间协作功能的接口，以便由多开发商合同开发的工作流产品，可以在不同的执行服务上执行过程的某一部分。采用这种方法，比要制定标准的内部接口、标准分布式执行服务状态数据要现实的多。

工作流客户端应用程序——可选择的实现方法

在工作流模型中，通过定义良好的、包含任务表的接口来实现任务表处理器与工作流机的交互，工作流执行服务器把任务队列分配给相应的用户（也可能是一组用户）。在最简单的情况下，工作流机访问任务表，进行任务分配；任务表处理器（工作流客户端应用程序）访问任务表，向任务表中添加任务。

任务表交互有许多不同的实现模型，这要依靠产品实现特性，尤其是用来支持任务表分配的低层组件类型来确定。

下图描述了**4**种实现方法，一种是支持集中式任务表处理的，三种用于分布式任务表处理的：



Alternative client worklist handler implementations

● 主机模型（**Host based Model**）——客户端任务表处理器应用程序是以主机为基础的，通过工作流中的本地接口与任务表通信。在此情况下，通过终端或者远程工作站的**MMI**（人—机界面）来驱动用户接口。

● 共享文件存储模型（**Shared filestore model**）——任务表处理器程序作为客户功能来实现，通过文件共享进行通信，共享文件存储位于主机与客户机平台环境的边界，能被二者访问。

● 电子邮件模型（**Electronic mail model**）——使用电子邮件系统进行通信，电子邮件系统可以把任务分配给过程参与者。此模型，任务表一般是存放在客户端的。

● 过程调用或消息传递模型（**Procedure Call or Message Passing model**）——通过过程调用或者其他消息传递机制进行通信。根据具体的实现特性，任务表可能放置在工作流机中，或者任务表处理器中。

在上边的每种情况中，都可以构造公共的**API**，支持任务表处理器访问任务表，支持工作流机的一些功能，但是公共的**API**要放在，适应产品实现特性的具体任务表访问功能后。

2.5. 对标准化的需要（The Need for Standardization）

由以下两个主要原因，驱使实现工作流重要功能接口的标准化：

- 继续支持过程重组和操作的灵活性
- 满足产品专业性与市场变化的需要

过程重组和操作灵活性（Business Re-engineering & Operational Flexibility）

业务过程重组与相应的工作流实现是有战略重要意义的，从而要求工作流产品要有足够的灵活性来处理不停（一直在）变化的业务过程，这是采用工作流技术的主要动机。有时，用不同工作流产品实现的几个单独的业务过程，后来业务过程重组为一个单一的复合过程，包括现有的工作流产品间进行交互。这种需求可能由于组织的合并、立法的变化、业务对象的改变等引起。随着电子数据交换的发展，工作流中也包含组织间通信，就象一个组织内通信。

在这些情况下，非常希望在不同的组织或者部门间使用不同的产品，而那些不能用于协调工作的产品，在过程变化时会产生十分严重的潜在问题。

标准化与市场变化（Specialization and Market Variety）

市场中有非常多的工作流产品，都针对不同方面的应用和不同的数据/应用程序集成。协同工作标准的制定，将使得我们可以从每一应用领域的产品中最好的来满足需求。这样的可以从一个开发商处选择过程分析、定义产品，而从另一开发商处选择工作流机软件，然后从第三个开发商处购买客户端任务表处理程序与前两个产品集成。

一个工作流，可以方便的拆分为几个子过程，每一个子过程由适合特殊数据类型、平台、网络环境的专门工具执行。协同工作流标准，提供了采用综合方法来适应业务过程的需要，把几个专门的工具的连接在一起满足过程的精确需求。

此外，许多工作流程序需要与其他系统集成，已有的或者将要出现的系统，从桌面办公系统到社团事

物处理。标准化的接口可以支持这种集成，并减小产品的复杂度。

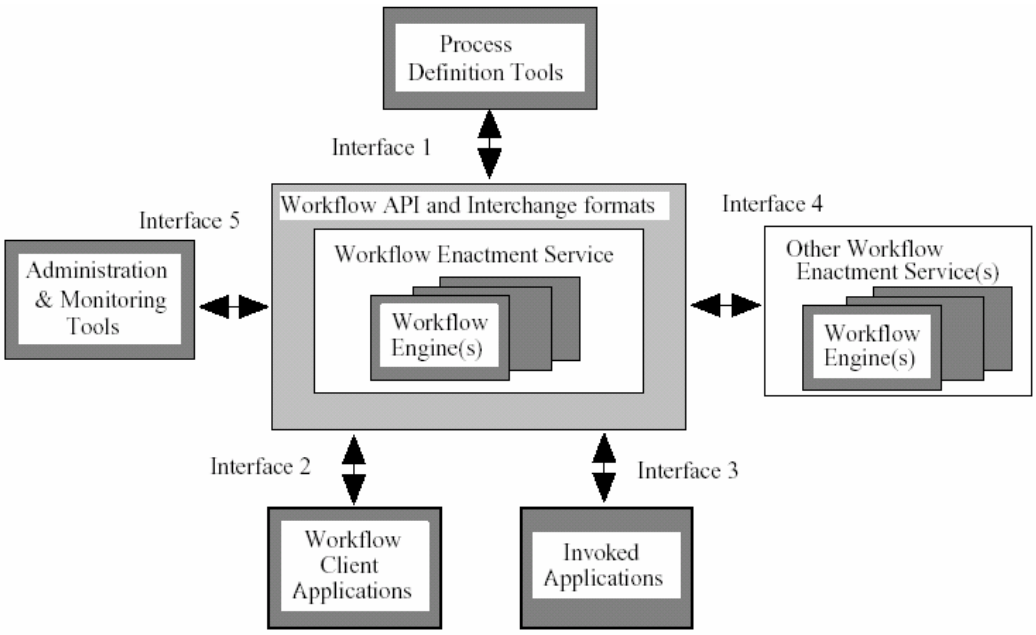
3. 工作流参考模型（Workflow Reference Model）

3.1. 简介

工作流参考模型来源于对普通工作流程序结构的分析，确定结构中的接口，这些接口可以使不同产品在不同的结构层次上协同工作。所有工作流系统都包含一系列的公共组件，组件间采用一套被定义好的方法进行协作；不同的产品在这些公共的组件中，会表现出不同的处理能力。为了实现不同工作流产品间的协同工作，需要在这些组件间制定一套标准的接口和数据交换格式。通过实现这些标准接口，可以达到产品间的协同工作。

3.2. 工作流模型（The Workflow Model）

下图描述了工作流体系结构中的主要组件与接口：



Workflow Reference Model - Components & Interfaces

工作流执行服务器周围的接口是**WAPI**（**Workflow APIs**），通过这些接口可以访问工作流系统的服务，这些接口还控制工作流控制软件与其他系统组件间的交互。在这**5**个接口中的许多功能，都是被**2**个或更多个接口同时拥有的，因此**WAPI**可以看作是统一的服务接口，可以交叉使用这**5**个接口来支持工作流管理功能，而不是单独的使用其中某个接口。

3.3. 工作流执行服务器（Workflow Enactment Services）

3.3.1. 什么是工作流执行服务器？

工作流执行服务器使用一个或多个工作流机，为过程实例和活动提供运行环境，负责解释和激活过程定义，与过程所需的外部资源进行交互。

> “工作流执行服务器”定义——
>
> 由一个或多个工作流机构成的软件服务器，用来创建、管理、执行工作流实例。应用程序可能会通过**WAPI**来与这个服务交互。

在模型中，过程与活动控制逻辑间有一个逻辑上的分离，活动控制逻辑构成工作流执行服务器；过程与应用工具间、与终端用户任务间也有一个逻辑上的分离，应用工具和任务建立起对每个相关活动的处

理。这种逻辑上的分离，为制定更多的行业标准提供了机会，也为在工作流程序中集成用户具体的应用工具提供了机会。

使用下边两个接口中的一个，就可以使工作流机访问外部资源：

● 客户端应用程序接口（**The client application interface**），通过这个接口工作流机可以与任务表处理器交互，代表用户资源来组织任务。然后由任务表处理器负责，从任务表中选择、推进任务项。由任务表处理器或者终端用户来控制应用工具的活动。

● 应用程序调用接口（**The invoked application interface**），允许工作流机直接激活一个应用工具，来执行一个活动。典型的是调用以后台服务为主的应用程序，没有用户接口；当执行活动要用到的工具，需要与终端用户交互，通常是使用客户端应用程序接口来调用那个工具，这样可以为用户安排任务时间表提供更多的灵活性。

在分布式的工作流执行服务器中，每个工作流机控制过程执行的一部分，并与这部分过程中的活动所要用到的用户、应用工具进行交互。在分布式的执行服务器中有公共的名称空间与管理范围的，从而过程定义、用户/应用程序的名称在一致的标准下被处理。分布式工作流系统，在工作流机间采用特殊的协议和信息转换格式，来同步工作流机的操作、过程交换和活动控制信息。也许工作流相关数据也要在工作流机间进行传递。在单一的工作流执行服务器中，这些操作都是由开发商自己定义的。

在工作流机间需要一个标准的交换格式，来实现异种产品间的调用。使用接口**4**，执行服务器可以把活动或者子过程转移到另外一个（异种）执行服务器中执行。在工作流参考模型中，这被称作“工作流机交互（**Workflow Engine Interchange**）”。

在异种环境中也需要公共的管理和监视功能，在**3.8** 讨论。

3.3.2. 工作流机（The Workflow Engine）

一个工作流机负责执行服务器中的部分（或者全部）运行控制环境。

“工作流机”——定义：

为工作流实例提供运行时期的执行环境的软件服务器或引擎。

工作流机能处理：

● 解释过程定义

● 控制过程实例—创建、激活、挂起、终止等

● 为过程的活动导航，可能要包含顺序或者平行的操作、最后时间期限、对工作流相关数据进行解释

● 参与者签名和退出

● 确定任务项目，实现用户意图；提供接口，支持用户交互

- 维护工作流控制数据和工作流相关数据，在应用程序间或者用户间传递工作流相关数据
- 提供调用外部程序的接口，连接所有工作流相关数据
- 提供控制、管理和审查功能

工作流机可以控制过程集、子过程、或通过对象类型的范围、及其属性定义好运行范围的实例。

在一个由多个工作流机构成的工作流执行服务器中，要把过程进行划分，分配给工作流机。可以按照过程类型来划分，某个工作流机负责控制相应类型过程；按照功能进行划分，某个工作流机负责控制过程的一些部分，这些部分所需要的用户或者资源，都在此工作流机的控制范围内。也可以按照其他的一些机制来划分。

3.3.3. 同种 和 异种的工作流执行服务器（Homogeneous & Heterogeneous Workflow Enactment Services）

同种工作流执行服务器由一个或多个兼容的工作流机组成，工作流机为工作流过程提供运行时期的执行环境。在多个工作流机间组织过程执行的机制、协议和转换格式，可以是产品所特有，并不一定是标准化的。

异种工作流执行服务器是由两个或者多个同种的执行服务器组成，并在一致性级别上遵守公共的协同工作标准。因此也需要定义一系列的一致性级别，来支持不断增加的公共功能级别。

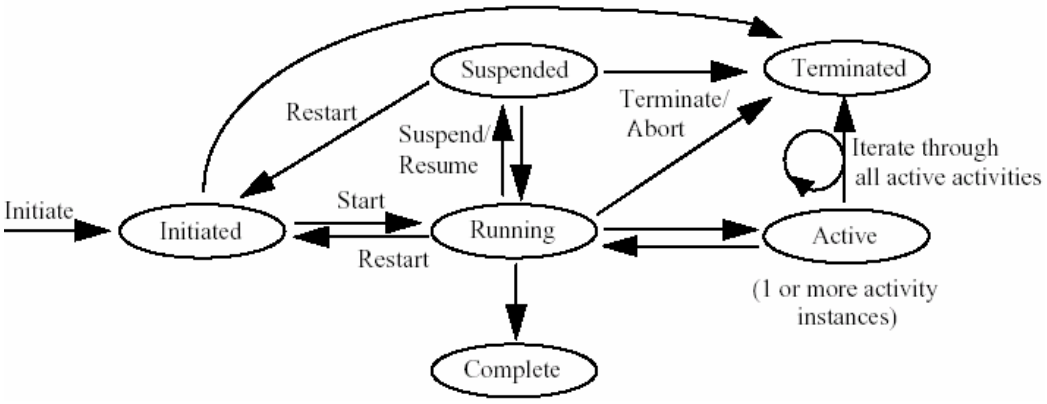
一致性级别中应该包括：

- 在异种执行服务器间要有公共的命名方案
- 异种执行服务器都要，支持公共的过程定义对象和属性
- 支持工作流相关数据在异种执行服务器间的传递
- 支持过程、子过程或者活动在异种工作流机间的传递
- 支持公共的管理和监视功能

过程和活动状态变迁（Process and Activity Transitions）

工作流执行服务可以看作是一个状态变迁机器，过程或者活动的实例在响应外部事件、工作流机负责的控制判断后，其状态发生改变。

下图描述了过程实例的基本状态变迁方案：



在上图中，发生状态转移（用箭头表示）来响应**WAPI**的命令；过程定义中的转移条件满足，也可能发生状态转移。

Initiated（初始化） —— 过程实例被创建，包括与过程状态相关的日期、工作流相关数据，但是过程还没有满足条件，不能执行。

Running（运行） —— 过程实例已经执行，过程中的活动如果条件满足就可以执行。

Active（激活） —— 过程中的一个或者多个活动已经被执行。

Suspended（挂起） —— 过程实例被静止，并且过程中的活动不能执行，直到过程返回到**running**状态。

Completed（结束） —— 过程实例满足结束条件；所有的完成后操作都将被执行（例如记录日志、或者统计信息），并且销毁过程实例。

Terminated（终止） —— 过程实例在正常结束前被停止；所有的完成后操作都将被执行（例如记录错误信息、或者恢复数据），并且销毁过程实例。

活动是不能被中断的，例如工作流执行服务器一旦开始了一个活动，就不能挂起或者终止这个活动。这就意味着，只有在所有运行中的活动结束后，并且过程返回到**running**状态，才能对过程执行挂起、重启、终止等命令。另外，可能需要把几个活动放在一起作为“原子单元”，这些原子单元要执行就全部被执行完，如果中途出现异常则返回到开始点，重新执行。可中断活动的处理办法和原子活动单元的重新启动能力，需要进一步的考虑，这超出了**WFMC**的初期工作范围。

忽略那些额外的复杂事物，活动实例的基本状态和转移如下图：

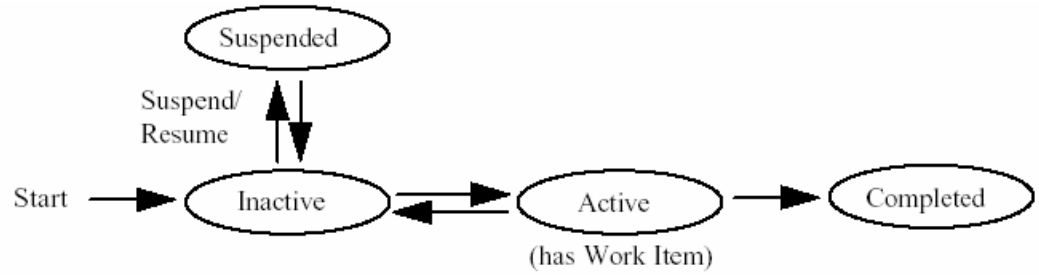


Figure 8 - Example state transitions for activity instances

一个活动的基本状态有：

Inactive —— 过程实例中的活动已经被创建，但是还没有激活（例如，活动的进入条件没有满足），并且没有任务需要处理。

Active —— 创建好的任务，分配这个活动来处理。

Suspended —— 活动实例被静止，并直到活动返回到**Inactive**状态，才能为其分配任务。

Completed —— 活动实例执行完成。

当然，一个产品也可以实现一些其他的状态类型，或者使用不同基本状态和转移来代表上图中的状态和转移。参考模型没有指定工作流系统的内部行为，但是状态转移阐明了，**API**命令集的影响范围的基本观点。

3.3.4. 工作流应用编程接口与数据交换（Workflow Application programming Interface & Interchange）

WAPI可以被看作是一套由工作流执行服务器支持的**API**调用和数据交换集合，这个集合在在工作流执行服务器的边界处，负责与其他资源交互。尽管结构中涉及到了**WAPI**中的**5**个接口，但是每一个接口中的许多功能都是公共的（即，同时被**2**个或者多个接口共同拥有）。

WAPI的主要功能由**API**调用组成。同时在**WFMC**也定义了接口间的，数据转换格式，例如过程定义。**WAPI**的使用在下边的部分进行讲解。

3.3.5. 工作流控制，工作流相关数据和工作流应用数据

工作流执行服务器维护内部控制数据，来确定过程实例或活动实例的状态，并支持其他内部状态信息。这种内部控制数据不能被访问，也不能进行转换。但是有些信息内容是要对外提供的，来响应某些特

殊操作（例如，查询过程状态等）。同种工作流执行服务器可能在工作流机间交换这些信息，通过使用具体的内部对话。

“工作流控制数据”——定义：

由工作流管理系统和（或）工作流机管理的内部数据。

工作流管理系统使用工作流相关数据来判断转移条件是否满足，并选择下一个要执行的活动。这些数据能被工作流应用程序访问，这些数据也需要通过工作流执行软件在活动间传递。当在同种环境下进行操作时，如果过程的执行要在2个或者多个工作流中进行，那么这些数据就要在工作流机间进行传递；这个过程可能需要名称映射或者数据转化。

“工作流相关数据”——定义

工作流管理系统用来判断过程中状态转移是否可以执行的数据。

过程实例中的每个活动中可能都需要进行数据操作。因此，工作流模型必须能够所有的处理活动间的“情形数据”交换。在一些环境中，可能需要情形数据在不同的工具数据格式间进行转换，例如，把文档从一种格式转成另外一种格式。（有的系统中，数据转换是工作流执行服务器来完成的；有的系统中，直接把数据转换定义成过程中的一个活动来执行）

“工作流应用程序数据”——定义：

应用程序的具体数据，并且不能被工作流管理系统访问。

工作流应用程序数据不能被工作流执行软件所使用，只与应用程序或者用户任务的执行相关。就象工作流相关数据一样，在同种执行服务器中应用程序数据会在工作流机间进行传递，来保证活动的正常执行。

应用程序与其需要用到的工作流相关或应用程序数据间的关系，会在工作流定义中说明。在一些情况下，可能是隐含关系（例如，在一些系统中情形数据会作为活动导航的一部分，传递到下一个活动中），然而在其他情况下（例如访问共享对象存储），就需要明确定义对象的名字和应用程序的访问路径。在参考模型中，把前一种情况称为“直接数据交换”，后一种称为“间接数据交换”。

3.3.6. 数据交换（Data Interchange）

工作流相关数据和应用程序数据的交换，都需要访问WAPI，来支持在3个运行时期功能中的协同工作：

- 任务表处理器（Interface 2）
- 应用程序调用（Interface 3）
- 工作流机交换（Interface 4）

本节讲述数据交换的基本原理；提出了API命令集，包括从工作流机中接收/返回工作流相关数据的具体调用；并为直接数据交换和间接数据交换定义了，与上述API命令集不同的命令集。

由Email驱动的工作流系统是一种典型的，应用程序数据的直接交换，这这样的系统中，应用程序数据物理地在活动间进行传递。这此情况下，不需要明确定义活动与应用程序数据间的关系；应用程序数据作为标准工作流活动导航的一部分进行传递，并且在应用程序调用时在本地直接与程序相关。需要在活动间提供数据格式转换时，应用程序需要定义与之相关的数据类型，可以作为一个属性来定义（这个属性信息可能存放在软件执行环境中，或者能被整个工作流执行服务器访问，例如地址目录）。这样，使用同种工作流应用程序构造的系统，就能够根据每个应用程序所定义的数据类型进行数据转换。需要采用一些协定来传递和保存数据类型信息，例如使用 X.400 对象标识符，或者Internet mail MIME机制。

一些类型的工作流系统（例如，使用共享文档存储实现的），在活动间不能从物理上传递应用程序数据。在这些系统中，应用程序要使用适当的访问路径才能进行数据访问。这样，必须要有统一的访问路径命名方案，必须是有效的访问权限，并且由激活（Active）的过程实例来控制访问权限。在这种情况下，如果需要，在建模时，数据格式转换也可以作为一个活动。

同种系统中可能使用私有的对象命名协定和访问权限，但是异种系统需要一个公共的方案。在异种系统中，在过程定义时必须包含对应用程序数据对象存储的访问路径，或者在活动间的导航必须包含访问路径的传递。

同种工作流产品进行协调工作，其必须采用相同的应用程序数据交换方法，或者通过一个网关机制进行协作，网关机制通过适当的协定，可以在两种不同的数据交换方法间进行映射，也可以处理对象命名与数据类型转换的不同。以后还需要对这部分进行细化，但有可能制定一个交换标准，来包含上述的两种情况。

工作流应用程序或相关数据交换的方法，都是通过3个接口来处理的；下边列出了这3个接口：

- 客户端应用程序接口（Client applications） —— 工作流相关数据可以包含在任务中。工作流相关数据也可以通过共享的对象存储形式来间接传递。
- 应用程序调用接口（Invoked applications） —— 依靠应用程序调用接口进行数据转换，可能需要在调用服务中把数据包含在具体应用程序协议中。激活的工作流应用程序可以使用，读/写工作流相关数据的API，或者用这些API来构造通用应用程序代理。
- 工作流机协作接口（Workflow Engine Interoperability） —— 与客户端应用程序接口相似，尽管在不同的系统中支持不同的应用程序数据交换方法，但是网关功能的使用，需要在两种方法间进行影射，也要处理名称问题。

3.4. 过程定义（Process Definition）

3.4.1. 过程定义工具（Process Definition Tools）

有许多不同的工具可以用来分析、建模、描述业务过程；这样的工具有很大的不同从非正式的（铅笔和纸）到成熟的、十分专业的。工作流模型不关系这些工具的特性，也不关心在过程建立时期他们是如何交互的。在以前指出过，这些工具可以作为工作流产品的一部分来提供，或者一个单独的产品，例如BPR工具集。

有的工作流产品提供了其自己的过程定义工具，从而过程定义一般是保留在 工作流产品范围内的，并且可能或者不能被，读/写信息的编程接口所访问。而使用单独的过程定义和执行服务器产品，过程定义能够在不同的产品间进行转换，并可以被其他产品访问。

设计活动和最后的过程模型输出，称为过程定义。在运行时期过程定义可以被工作流机解释。

过程分析工具、建模工具和定义工具，都要有在一个组织结构中模拟过程的能力（尽管这不是工作流参考模型规定必须有的）。如果组织模型集成到了这些工具中，那么过程定义将包含组织相关对象，例如角色。这些都是与系统相关的控制数据，例如角色： 活动者间的关系，可能会在过程执行期间被引用。

3.4.2. 工作流定义转换（接口 1）

在建模或定义工具与运行时期工作流管理软件间的接口，被称为过程定义导入/导出接口。这个接口的特点是：转换格式和API调用，从而支持过程定义信息间的互相转换。这个接口也支持已完成的过程定义间的互相转换，或过程定义的一部分。例如，过程定义的改变或者活动中属性的改变。

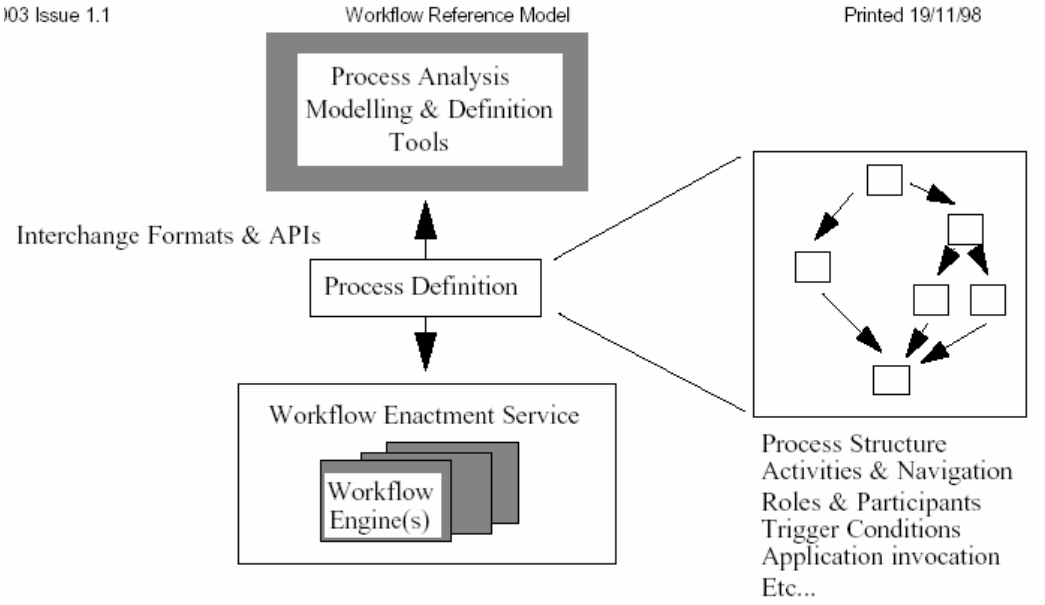


Fig 9 - Process Definition Interchange

使用标准的过程定义格式有很多好处：

首先，把建立阶段与运行时期环境进行了分离，可以使用一个建模工具来产生过程定义，这个过程定义可以作为很多个不同工作流运行时期产品的输入。从而用户可以单独地选择建模工具和工作流运行时期产品。

其次，可以为几个工作流机输出过程定义，这几个工作流机合作来构成分布式的工作流执行服务器。

WFMC在此部分作了以下两个方面的工作：

① 提出了一个元模型，可以用来表示过程定义中的对象、对象间的关系和属性。这个元模型为不同的产品间的过程定义相互转换奠定了基础，并形成了一套转换格式。

② 工作流系统间或工作流系统与过程定义产品间的API调用，提供了公共的方法来访问工作流过程定义。访问可能是读、读/写或者只写操作，并且操作标准对象集合（在元模型中定义的对象集合），或者产品自己的对象集合。

基本元模型（A Basic Meta-Model）

WFMC开发了一个过程定义的元模型。元模型中定义了基本的对象类型集，来满足简单的过程定义相互转换。或者有开发者具体扩展，或者在增加的功能中定义另外的一直性级别来增加更多的对象类型。

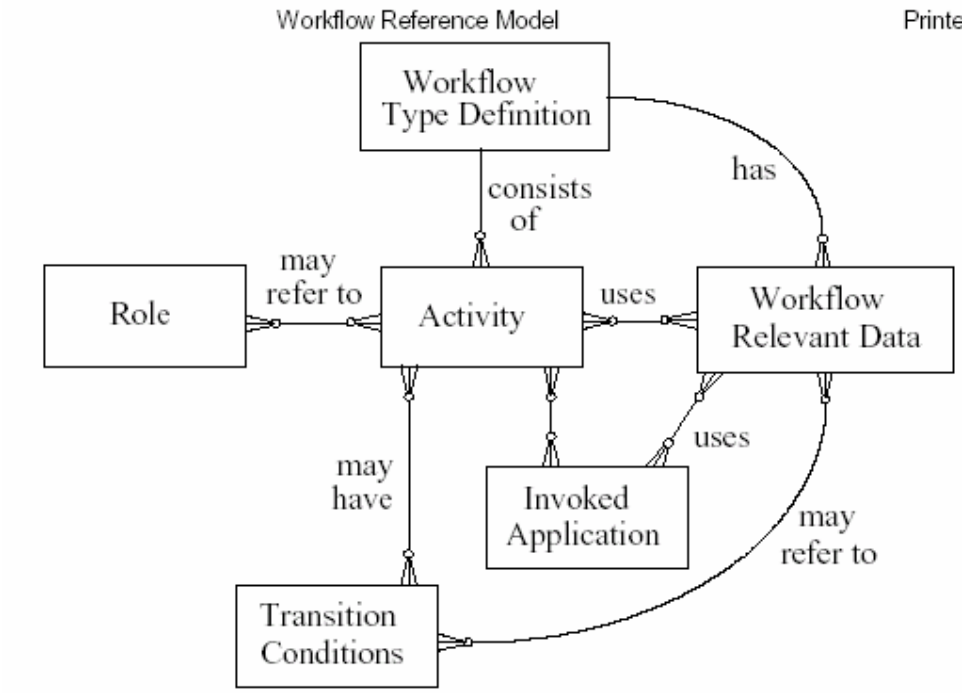


Fig 10 - Basic Process Definition Meta-model

需要为下边的类型定义特殊的属性：

工作流类型定义（Workflow Type Definition）

- 工作流过程名
- 版本号
- 过程开始/结束条件
- 安全、审查、控制数据

活动（Activity）

- 活动名
- 活动类型
- 进入动作和离开动作
- 其他约束

转移条件（Transition Conditions）

- 执行条件

工作流相关数据（Workflow relevant data）

- 数据名与路径
- 数据类型

角色（Role）

- 名称与组织实体

应用程序调用（Invoked Application）

- 类型和名称
- 执行参数
- 本地或者访问路径

在分布式工作流服务器中，可能要在过程定义时，为每个工作流机分配活动，可以作为活动的一个附加属性。过程定义能影响安全性与管理。

定义的交换格式，要支持符号命名方案，这些符号可以映射到工作流执行服务器中的实际名称与地址。这种映射可以使用动态地址定位机制来实现（例如，目录服务器），也可以使用其他的外部过程定义机制实现。也有其他的一些行业在相关的方面作研究，例如过程建模和CASE转换工具；WFMC提出的方法也适用与其他行业，预先定义适当的转换格式。

访问过程定义的 API（APIs to access Process Definitions）

用来支持访问过程定义数据的API命令集。希望规范中包含下边列出的通用类型功能。命令集应该提供命令操作表，和操作的对象、属性。

建立会话（Session Establishment）

- 连接/断开参与系统间的会话

工作流定义操作（Workflow Definition Operationis）

- 从过程定义库或者其他资源中，获得工作流过程大鲵故意的名称列表
- 选择工作流过程定义，为更多的对象级操作提供会话句柄
- 读/写上层工作流过程定义对象

工作流定义对象操作（Workflow Definition Object Operations）

- 创建、恢复、删除工作流定义中的对象
- 恢复、设置、删除对象的属性

3.5. 工作流客户端功能（Workflow Client Functions）

3.5.1. 工作流客户端应用程序（Workflow Client Applications）

任务表处理器是在需要调用人类资源的活动中，用来与终端用户进行交互的软件。任务表处理器可以作为工作流产品的一部分提供给用户，也可以由用户自己开发。在其他情况中，工作流可能要与普通的办公系统进行集成，例如Email，来为终端用户提供一个统一的任务管理系统。这就要求在工作流执行服务器与工作流客户端应用程序间有一个非常灵活的通信机制，来构建各种可能遇到的运行系统。

在工作流模型中，通过客户端应用程序与工作流机间的定义良好的接口进行交互。在这个接口中包含任务表——由工作流机分配给用户的任务序列。最简单的情况是，工作流机访问任务表，来把任务分配给用户；任务表处理器访问任务表，向任务表中添加任务项。有许多不同的产品来实现任务表的交互。

任务表中任务项的激活（例如，启动应用程序，连接工作流相关数据），可能是由工作流客户端应用程序或者终端用户控制的。在工作流客户端应用程序与工作流执行服务器间定义了一系列的方法，用来向任务表中添加任务项、从任务表中删除完成的活动、激活临时挂起的活动，等。

任务表处理器也可以调用应用程序，或者直接调用，或者由终端用户调用。通常希望，任务表处理器的应用程序调用范围能够受到运行环境的限制，尽管这样会给模型带来通用性的限制，但这种情况是一直存在的。

与任务表相关的部分活动的数据，是任务表处理器用来调用应用程序所必须的信息。当应用程序数据是强类型时，在任务表处理器中要存放一个联接，用来实现程序的调用。在其他情况中，在任务表处理器与工作流机间要进行完全的应用程序名称和地址信息的交换；这时，工作流客户端应用程序也可能实现一些应用程序调用接口（接口**3**）中的功能，来获得必要的信息。

任务表中可能要包含一个过程中的几个不同实例的相关任务，或者包含几个不同过程中的一个共同活动项。一个任务表处理器可能要与几个不同的工作流机、几个不同的工作流执行服务器进行交互。（按照每个产品的实现，为每个过程单独维护一个物理上分开的任务表，或者任务表处理器把几个不同的任务表联合到一起，呈现给终端用户）

因此，客户端工作流应用程序与工作流机间的接口必须十分灵活，来满足下边的几方面功能的实现多样性：

- 过程和活动表示符
- 资源名和地址
- 数据引用和数据结构
- 可选择的通讯机制

3.5.2. 工作流客户端应用程序接口（接口 2）

满足上述需求的方法，在标准**API**集后，可以为从工作流应用程序到工作流机和任务表的访问提供一致的形式，而不管产品的实现特性。

API与其参数可以映射到几个不同的通信机制上，来适应各种不同的工作流实现模型。

下图是对客户端应用程序**API**方法的一个总揽：

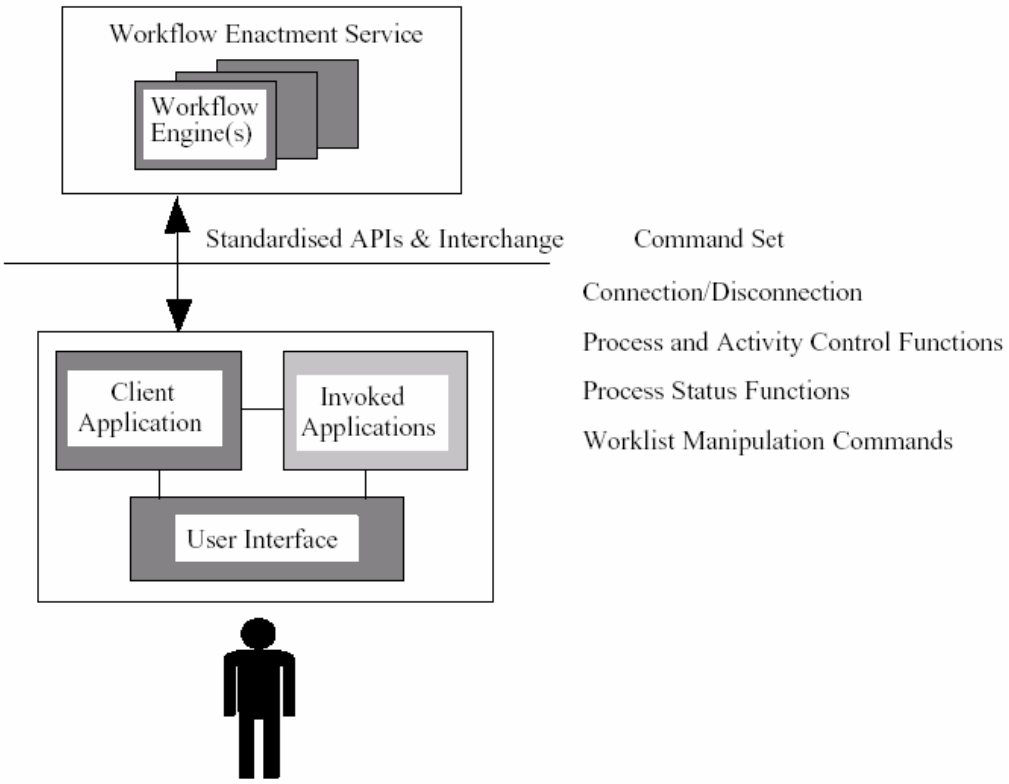


Figure 11 - Client Application Interface

WFMC在其分档中，分开发布了**API**规范；下边是对客户端应用程序**API**使用的一个概述，分成几个不同的功能。提供了对单独或者多个过程活活动实例的操作命令，就象任务表一样。

建立会话（Session Establishment）

- 连接/断开参与系统间的会话

工作流定义操作（Workflow Definition Operations）

- 对工作流过程定义名称或者属性的恢复/查询功能

过程控制功能（Process Control Functions）

- 创建/开始/结束一个过程实例
- 挂起/唤醒一个过程实例
- 在过程实例或活动实例中强制一个状态发生改变
- 查询过程实例或活动实例的属性

过程状态功能（Process Status Functions）

- 打开/关闭过程实例或活动实例的查询，设置过滤标准
- 获取过程实例或活动实例的详细信息
- 获取具体过程或活动的详细信息

任务表/任务项处理功能（Worklist/Workitem Handling Functions）

- 打开/关闭任务表查询，设置过滤标准
- 获取任务表中的项目
- 通知选择/重分配/结束一个任务项
- 查询任务项属性

过程管理功能（Process Supervisory Functions）

- 改变过程定义或者它的实例的运行状态
- 改变某种类型的所有过程实例或活动实例的状态
- 为某种类型的所有过程实例或活动实例的属性赋值
- 终止所有过程实例

数据处理功能（Data Handling Functions）

- 恢复/返回工作流相关或应用程序数据

应用程序调用（Application Invocation）

- 上边对功能的概括，为支持任务表处理器对应用程序调用提供了基础。应用程序调用功能的一些命令是与客户端应用程序环境相关的。

- 有些产品可以只实现全部**WAPI**的一部分；以后会给出进一步的考虑，定义一致性级别，来满足市场中不同的产品间的，不同的协作需要。

3.6. 应用程序调用功能（Invoked Application Functions）

3.6.1. 应用程序调用（Invoked Applications）

所有的WFM产品都没有足够的逻辑单元，知道如何调用所有的应用程序，这些应用程序存在异种的产品环境中。这就需要，能够处理在所有平台下和网络环境中进行调用的逻辑，并需要能使用公共格式和编码进行应用数据或相关数据传递的方法。

然而，许多工作流系统能够使用了更多受限制的应用程序，特别是那些采用强制数据类型和直接与应用程序相连的系统。在其他情况中，应用程序对操作的调用，可能是通过边准的交换机制来实现的，例如OSI TP协议或者X.400。一些实现使用了“应用程序代理（Application Agent）”，把这些在在标准接口之后的各种方法包含在工作流执行服务器中。也有可能开发“Workflow enabled”应用工具，这种工具使用标准的API集来与工作流执行服务器进行通信，来接收应用程序数据、信号和响应活动事件，等。这些API可以被应用工具直接调用；也可以被应用程序代理过过程调用，作为与其他应用程序（不包含任何工作流技术的程序）交互的前端。

下表中列出了应用程序调用用到的一些接口类型：

接口类型	工作流相关数据访问	候选标准
本地过程调用（Local Process Call）	本地文件	无
Shell 脚本（Shell Script）	本地文件	POSIX环境？
ORB调用（ORB Call）	通过引用（调用参数）	有
远程执行调用（Remote Execution Call）	通过引用（调用参数）	有
消息传递（Message Passing）	内含或引用（Embedded or via reference）	有
事务处理（Transaction）	内含或引用（Embedded or via reference）	有

3.6.2. 应用程序调用接口（接口 3）

下边是接口3的结构， “workflow enabled”类型的应用程序或应用程序代理，可以直接使用这个结构。

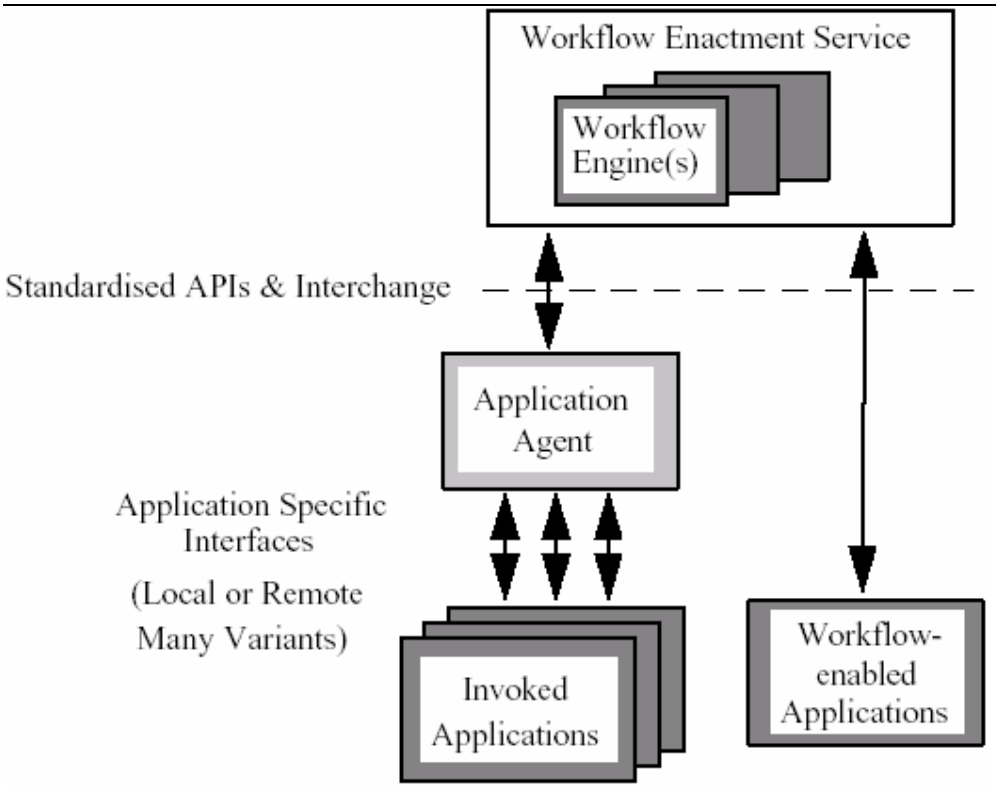


Figure 12 - Invoked Application Interface

在简单的情况中，工作流机在本地处理应用程序调用，使用过程定义中的信息来确定，活动的性质、将要调用的应用程序的类型和所需的数据。被调用的应用程序可能存储在工作流机中，或者与工作流机一同存储在相同的平台下，或者存放在一个独立的网络访问的平台中；过程定义中有足够的应用程序类型和寻址信息（工作流机的特殊需求），来实现应用程序调用。在这种情况下，应用程序命名与寻址的协定是处于，工作流机与过程定义之间的。

应用程序调用**API**的详细语法、语义作为**WFMC**规范的一部分给出。操作覆盖了一些不同的基本接口，包括上表中的一部分，其中一些操作是同步的，一些是异步的。**API**的操作可以是单线程的，也可以是多线程的，后者使用活动**ID**来区分线程。下边是应用程序调用可以使用的一些命令概括：

创建会话（Session Establishment）

- 连接/断开应用程序会话

活动管理功能（Activity Management Functions）

- 开始活动
- 挂起/恢复/放弃 活动
- 活动完成通知
- 信号事件

- 查询活动属性

数据处理功能（Data Handling Functions）

- 提供工作流相关数据
- 提供应用程序数据或数据地址

更复杂的情况，异种工作流机间的协同工作，可能需要在工作流机间传递应用程序调用信息，或者作为运行时期数据交换的一部分，或者通过在过程定义阶段后导入过程定义来实现。

3.7. 工作流协同工作流能力（Workflow Interoperability）

3.7.1. 异种工作流执行服务器

WFMC的一个主要目标是，为不同开发商的工作流系统产品，相互间能够进行无逢传递任务项，定义标准。

工作流产品的特性变化多样。在WFMC的协同工作标准中，没有强迫开发商必须在①提供一个只面向用户需求的产品②只考虑协同工作，二者中作选择。

WFMC把焦点聚集到，开发多种不同的协同工作框架，这些框架可以操作一系列标准的协调工作，从简单的任务传递到整个工作流系统的协同工作(包括过程定义转换、工作流相关数据交换、通用的界面等)。简单的协同工作，WFMC的协同工作定义将在最初就能支持；而复杂的协同工作，还需要进一步的研究。

尽管可以开发一个非常复杂的协同工作框架，由许多个工作流机构成个执行服务器，但是这种框架不会在近期实现，因为这需要所有的工作流机都可以解释一个公共的过程定义和共享公共的工作流控制数据集，事实上是维护异种工作流机间的一个共享过程视图。现阶段更现实的目标是，能够在运行时期传递过程的某些部分，来支持不同的执行服务器运行。

WFMC定义了4个协同工作模型，包含多种协同工作能力级别。下边就来描述这几个模型。图中使用正方形来表示任务或者活动，用不同的阴影来区别每个工作流执行服务器的任务。

3.7.2 模型 1 —— 链锁式（Chained）

这个模型中，过程A中的一个连接点，连接到过程B中的一个连接点。尽管在图中给出了过程的开始连接点与结束连接点，但这并不仅是为了图解的目的。连接点可以是过程中的任何一个活动。

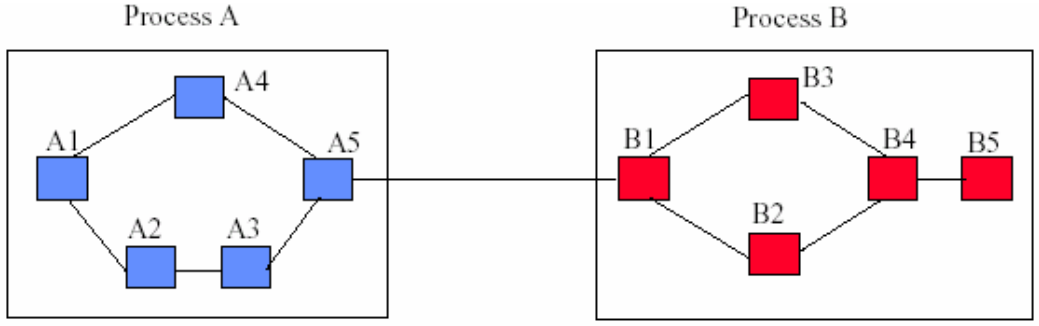


Figure 13 - Chained Services Model

这个模型支持在两个工作流环境中传递一个任务，这个任务会在第二个环境中独立执行，不需要同步。可以使用一个网关应用功能来实这种模型，网关负责处理数据格式转换、过程和名称映射等，例如在两个执行服务器中使用标准**API**调用。

3.7.3. 模型 2 —— 子过程嵌套（Nested Subprocesses）

一个过程可以在全部在一个工作机范围内执行，封装成父过程的一个任务，在一个与执行父过程不同执行服务器中运行。在父过程与封装过程间存在一个等级关系，构成父过程的子过程。这种等级关系可以延伸到很多层，形成嵌套子过程。产品可以实现回归，也可以不实现。

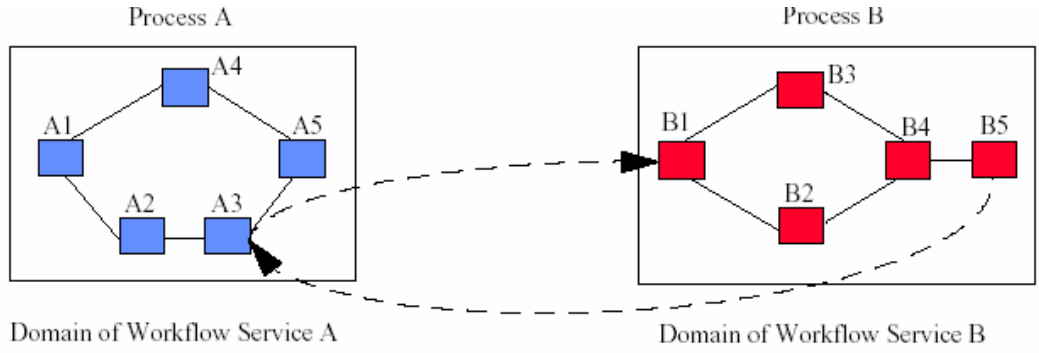


Fig 14 - Nested Subprocesses Model

在图中，工作流执行服务器**A** 有一个活动（**A3**），会在过程**B**结束后执行，过程**B**在工作流执行服务器**B**中执行，并在过程**B**执行结束时，执行服务器会把控制返回给执行服务器**A**。 在模型 **1** 中，活动控制的传

递，要经过两个工作流执行服务器间的应用程序网关功能或者直接的API调用。上图中描述的是一种简单情况，在过程B中只有一个单入口和单出口点，然而在B中的活动导航规则可能会使其他活动也满足此情况，例如在活动B5前过程完成条件就得到了满足，并且控制返回给了工作流A范围。

3.7.4. 模型 3 —— P2P （Peer-to-Peer）

这个模型是一个完全的混合环境；图中指示出了一个复合过程C，C中的活动需要在多工作流机执行服务器中执行，形成一个共享范围。活动C1、C2和C5由执行服务器A执行，活动C3、C4和C6由执行服务器B执行。

在这个模型中，过程是通明地由任务到任务来推进的，不需要用户或管理员参与，交互只在工作流机间存在。

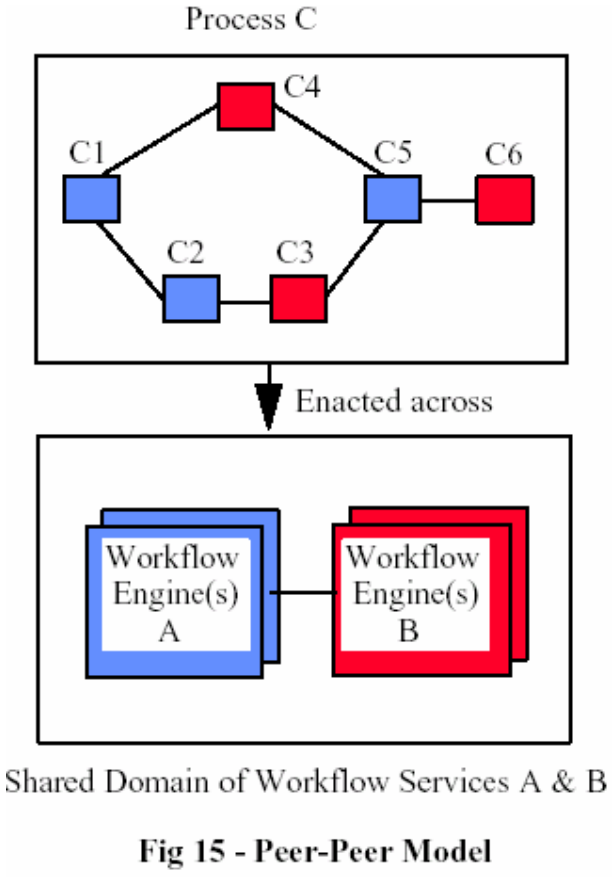


Fig 15 - Peer-Peer Model

在此模型中，需要两个工作流执行服务器都必须支持公共通信API集，并且两者都能解释一个公共的过程定义，或者从一个公共的建立过程引入到这两个执行环境中，或者在运行时期从两个服务器中导出。工作流相关和应用程序数据，也要能在多个异种工作流机间进行传递。

上图只是一种简单情况，P2P模型更复杂的情况需要进行进一步研究。如上图所示，活动与一个特殊

的工作流执行服务器相连（例如在过程定义中就进行了说明）。系统管理、安全性和恢复，需要几个工作流执行服务器的一起来完成。一种极端的情况，两个不同的工作流执行服务器要共享过程状态信息，而这些状态信息一般是保存在每个过程内部的，事实上这就形成了一个单异种执行服务器。WFMC将要定义一系列的一致性级别，允许早期规范能够处理简单的情况，并附加一些功能来满足复杂情况的需要。

3.7.5. 模型 4 —— 相似同步（Parallel Synchronised）

在这个模型中，允许在本质独立执行两个过程，可以反问独立的执行服务器，但是在两个过程间要有同步点。同步需要，一旦每个过程都达到在其执行序列中的一个预先定义的点，就要激发一个公共事件。这种机制可以用来在交叉执行的线程中进行过程时间安排、检查恢复数据、在不同的过程实例间传递工作流相关数据，等。

在下图中，过程A中的A3与过程B中的B4是同步点：

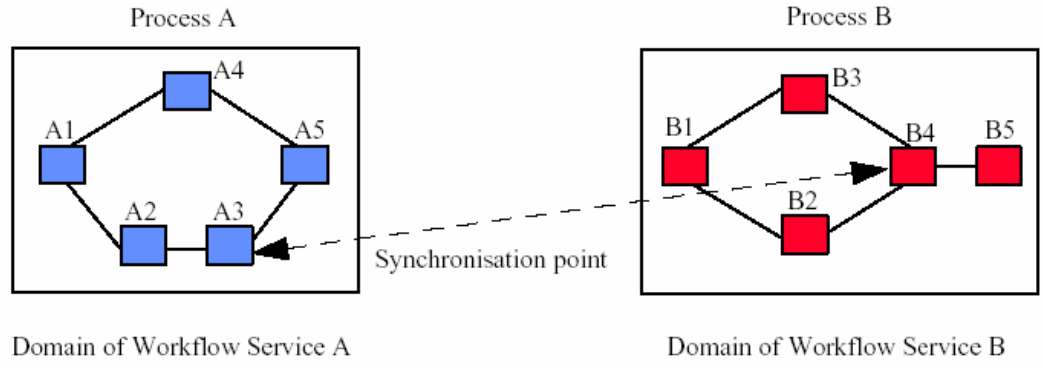


Fig 16 - Parallel Synchronised Model

一对匹配的任务，可以在每个过程中的某个点同步。这需要一个协调和跟踪机制事件，另外两个执行服务器都要能够从连个过程定义中识别任务。

3.7.6. WAPI 协调工作功能（接口 4）

异种工作流系统中信息与控制流的普遍性质，如下图：

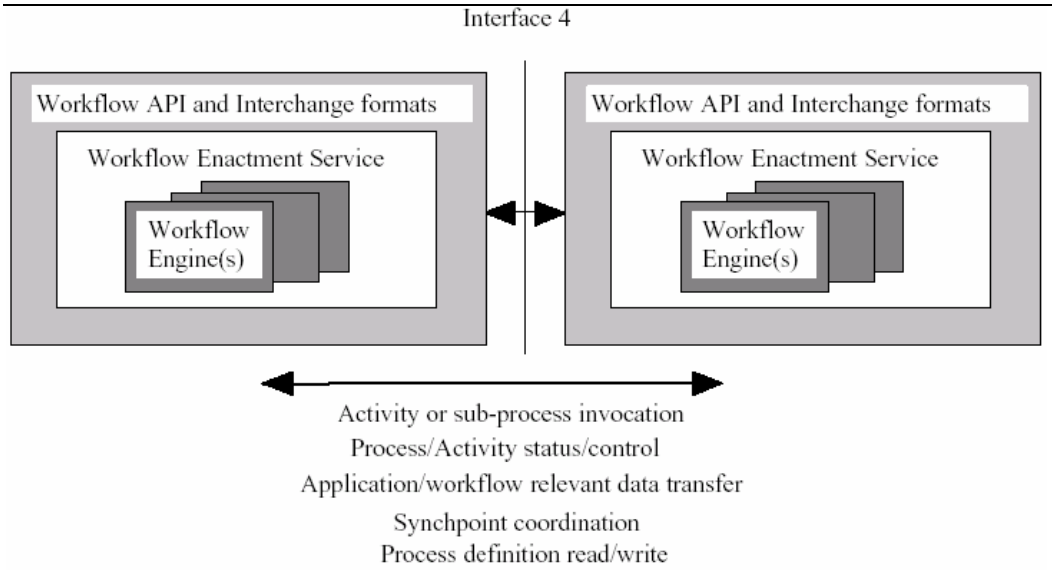


Figure 17 - Workflow interoperability interface

有两个主要的方面是必须协同工作的：

- 扩展对过程定义的公共解释，这是必须的，而且是可以实现的
- 在运行时期，支持多种类型的控制信息相互转换，在不同的执行服务器间传递工作流相关数据和应用程序数据。

在多个范围（工作流执行服务器）内使用过程定义

如果两个执行服务器能够解释一个公共的过程定义，这使得两个执行服务器共享过程定义对象和属性的视图。这个视图包括活动、应用程序、组织和角色名称、导航条件，等。每个工作流机要能够在公共的命名和对象模型上下文环境中，把活动或子过程传递给异种工作流机。这种方法特别适合协同工作模型**3**。

如果不能实现对过程定义视图的共享，可以采用“导出”过程定义的方法来代替。过程定义转换**API**提供了一个从工作流执行服务器中获得对象、属性等数据的手段，使得工作流机能够获得执行一个活动或者子过程所需要的过程定义数据。

如果过程定义转换不能由上述方法实现，就需要使用网关方法来实现协同工作。在网关方法中，使用一个应用协作网关，把对象名称、属性在两个执行服务器间进行映射。在最简单的情况下，两个分离的执行服务器使用他们各自的过程定义格式，然后通过网关进行映射。这种方法食用模型**1**与**2**，不适合模型**4**。

运行时期控制交互（RunTime Control Interactions）

在运行时期，使用WAPI调用在不同的工作流执行服务器间进行控制传递，在某个执行服务器中执行相应的子过程或者活动。如果两个执行服务器都支持公共级别的WAPI调用和公共的过程定义对象视图，那么控制传递将直接在工作流机间进行——尽管这需要公共的协议来支持WAPI。

如果信息不能直接在工作流间进行传递，可以使用WAPI调用来构造一个网关功能，把两个执行环境中或不同的协议环境下，的对象和数据视图进行映射来实现协同工作。如下图：

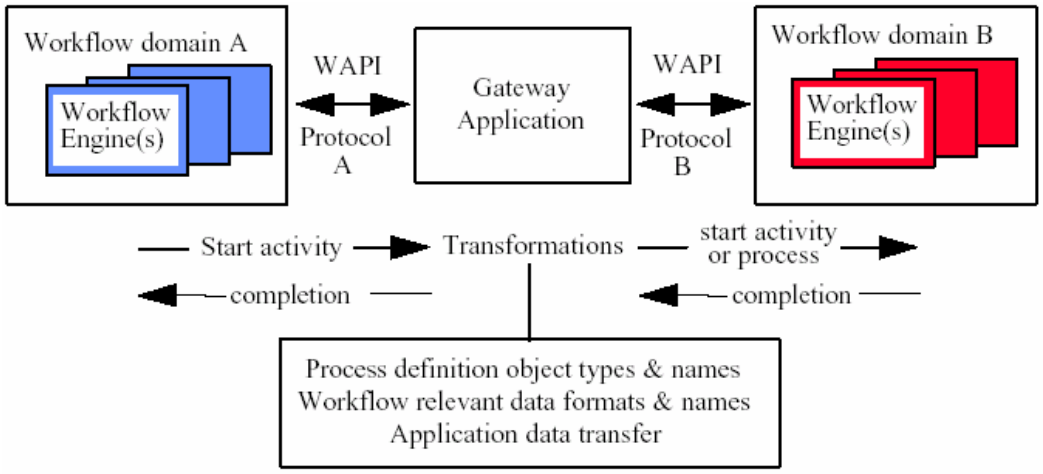


Fig 18 - Gateway operation using WAPI

上图中展示了网关运行的主要的原理；依靠具体的协同工作模型，一个活动可以从Domain （A）映射到Domain（B）中的一个活动或一个新的过程/子过程。

很多WAPI命令在根本上是支持协同工作的，或者通过在两个系统间的直接调用，或者通过网关功能来实现。许多已在前边（3.4.2，3.5.2 ，和3.6.2）讨论过的WAPI命令，也可应用于工作流协同工作中：

- 建立会话
- 对工作流定义及其对象的操作
- 过程控制和状态功能
- 活动管理功能
- 数据处理操作

在多工作流执行服务器间一定程度的公共管理功能，也需要使用接口5中的功能。

一旦活动被执行，工作流可户端应用程序与初始服务器间的交互（例如，查询活动/过程实例的状态，或者挂起/恢复/终止过程实例），可能会被推给下级服务器来进行。有些操作可能要连接起来，交叉于几

个工作流机中（例如，一个过程实例的多个活动，分布与几个工作流机中）。需要一些事件通知服务，来通知开始服务器活动/子过程的状态变化与完成等。同时也说明了，需要开发另外的一些WAPI操作，来支持更复杂的协同工作模型中的上述功能。

就状态转移来说，其交互的范围是相对广大的；将来需要开发必要的一致性级别，可以形成不同产品间协同工作的应用基础。

3.8. 系统管理（Systems Administration）

3.8.1 管理和监视工具（Administration & Monitoring Tools）

WFMC规范的最后关注的是，为管理和监视功能开发公共的接口标准，这样一个开发商的产品就可以用来管理其他工作流机的运行。通过公共的接口，几个不同的工作流执行服务器可以共享，管理和监视功能。

尽管，过程状态命令在接口定义中已经描述了，但一致认为，在某些行业中需要，进行全部状态监视和提取信息的功能。WFMC提出的接口，是要让用户能够得到工作流运行状态的完整视图，无论是什么样的工作流系统；同时，也希望能提供一套全面的功能集，进行系统管理，包括安全性、控制和权限。

接口中包含WAPI集中的一些具体命令，来操作管理和监视功能。另外，进一步的讨论，期望能够确定在什么范围内，这个接口可以使用现有的协议（如CMIP、SNMP），来设置、恢复管理状态和统计信息（定义在开放MIB中——Management Information Base）

3.8.2. 管理和监视接口（接口 5）

在下图中，一个独立的管理应用程序与多个不同的执行服务器进行交互。

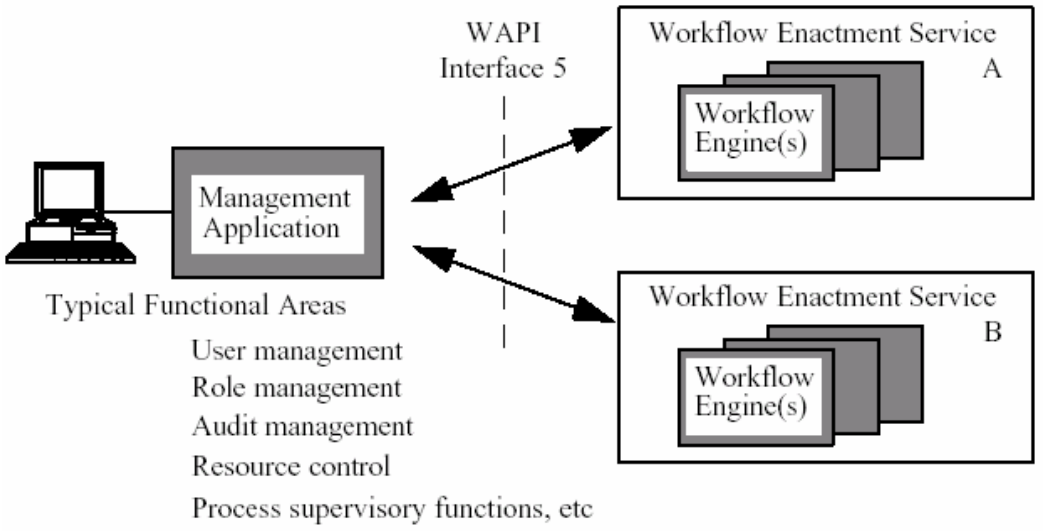


Fig 19 - Systems Administration & Monitoring Interface

除了上图中描述的外，管理应用程序可以执行其他的一些管理功能，例如可以管理工作流过程定义、作为资源库、通过接口1中的操作为各种不同的工作流执行服务器分配过程定义。

对此接口还需要进一步的研究，但此接口中至少应包含以下类型的操作：

用户管理操作（User Management operations）

- 建立/删除/吊销/修改用户或工作组的权限

角色管理操作（Role Management operations）

- 定义/删除/修改 角色的参与者
- 设置或取消角色属性

审查管理操作（Audit Management operations）

- 查询/打印/新建/删除审查记录或事件日志，等

资源控制操作（Resource Control operations）

- 设置/取消/修改 过程或活动并发级别
- 访问资源控制数据（数量、开始、使用参数等）

过程管理功能（Process Supervisory Functions）

- 改变工作流过程定义或其扩展过程实例的运行状态
- 使用/不使用 某个版本的过程定义
- 改变某一类型的所有过程/活动实例的状态
- 为某一类型的所有过程/活动实例的属性赋值
- 终止所有的过程实例

过程状态功能（Process Status Functions）

- 打开/关闭 过程/活动实例查询，设置过滤标准
- 取得过程/活动实例的详细信息
- 取得特殊过程或活动实例的详细信息

4. WAPI 结构、协议和一致性

4.1. WAPI——API 功能简介

WAPI是一套公共的API调用集，和支持5个功能接口互相协作的数据转换格式。在5个接口中定义了一些操作（在第3部分中已经讨论过），包括下边的几组操作：

API调用（API Calls）

- 建立会话
- 对工作流定义及其对象的操作
- 过程控制功能
- 状态功能
- 活动管理功能
- 数据处理操作
- 任务表/任务项 处理功能
- 用户管理操作
- 角色管理操作
- 审查管理操作
- 资源控制操作

数据交换功能（Data Interchange Funtions）

- 过程定义传递
- 工作流相关数据传递

API调用结构和命名

WFMC一开始就将定义API调用的逻辑操作和数据类型，API调用会操作、支持引用参数中的数据结构。期望开发绑定语言，首先会开发C语言，随后会为其他一些重要开发环境开发帮定语言（例如，C++、IDL）。也要为调用功能制定命名协定，并为数据类型定义、参数类型、数据结构制定命名协定。

4.2. WAPI 协议

WAPI调用会在如下的两类协同工作情况中起作用：

● 当一个WAPI外部接口，作为边界功能提供给工作流执行服务器时（例如，开发商的Stub程序包含在可户端应用程序或者程序代理中），需要使用开发商的具体映射机制，来对调用和相关参数进行编码，从而通过开发商自己的协议环境与工作流机进行通信。

● 要在不同的产品中直接进行协同工作，就需要公共的协议来支持。这将需要一个标准的映射，从WAPI调用到一个或者多个协同工作协议。

考虑到的与上述两种情况相关的标准，如下图：

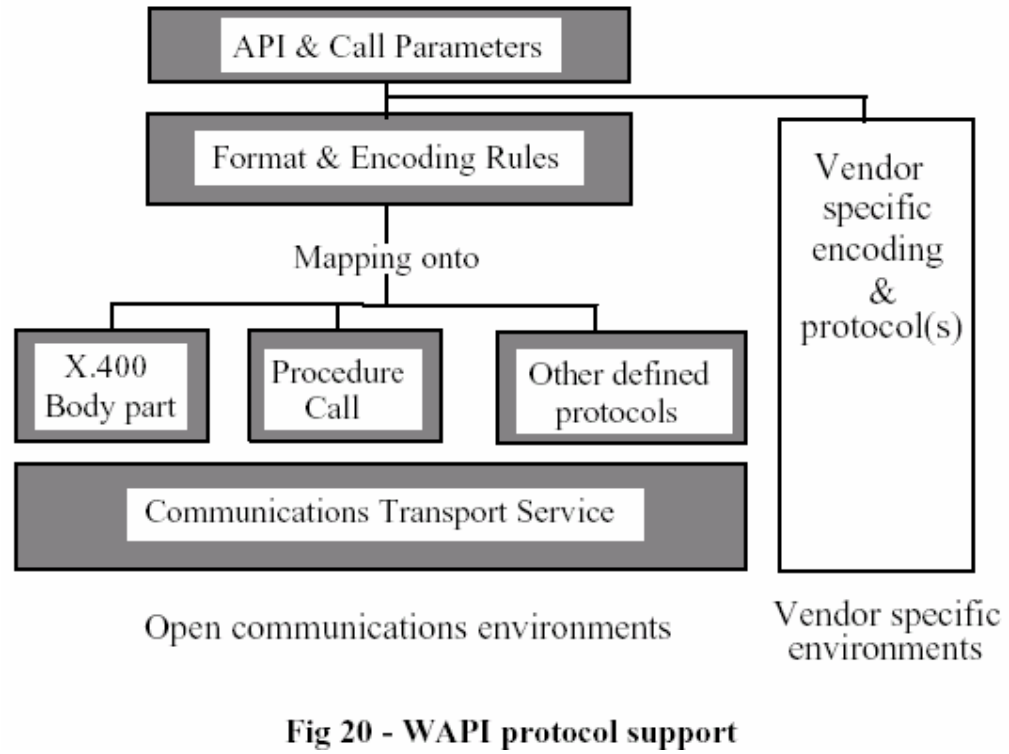


Fig 20 - WAPI protocol support

在WAPI中使用的协议需要进一步的研究，但是WAPI映射将要被开发为重要的通信环境，也就是那些在市场上的产品，所广泛使用的。使用网关（方法1）可以初步支持可户端应用程序集成和工作流协同工作；然而，这种方法有许多内在的限制，并且在中期需要开发适当的协议使用说明。

4.3. 一直性原则

4.3.1. 一致性的意义

将根据5个接口中的每个接口功能来定义一致性，从而开发商可以提供一个外部接口来构造一个或多个功能的协同工作，而不需要实现所有5个接口的功能进行协同工作。

每一个接口都希望，能把一致性划分几个级别，级别 1 提供一个最低协同工作级别，更复杂的产品可以实现更高一致性级别。在接口4中，定义一致性级别是必须的，因为会有更复杂的协同工作。实现某一致性级别的产品，可以与实现了同级或低级一致性的产品协同工作。

要单独考虑API支持、使用协议的一致性。可以构造一些矩阵，标志在某些一致性级别上支持的某些API功能，标志支持与其他产品协作的协议环境。

4.4. 协同工作能力分类和一致性级别

需要定义一致性级别，帮助对协同工作产品进行分类。

工作流系统协作和应用程序集成的范围是很大的，开发一个覆盖所有范围的API和交换格式来满足各种协作情况，是一个非常大的任务。因此，必须开发一套协作模型，从简单的到复杂的，以便可以先开发简单模型的接口。这样，简单的协同工作系统就可以在近期实现，同时继续开发更复杂的模型接口。可以定义不同的一致性级别，来对API、交换格式、支持协议进行分组，从而满足具体的协作模型。

下边是一个简单的分类情况，为进一步的讨论打下基础。

4.4.1 定义工具、工作流执行软件

目的 —— 可以单独选择产品作为开发工具（建模、定义等），运行时期工作流执行服务器可以发送或存储、恢复过程定义库中的过程定义。

协同工作能力 —— 以支持过程定义导入/导出的接口为基础。由定义工具导出过程定义；由工作流执行软件导入过程定义。

一致性级别 —— 以基本的过程定义对象集合为基础，并可以对其扩展以满足更复杂的过程定义。

4.4.2 可户端应用程与工作流执行服务器序协同工作

目的 —— ① 构造公共的任务表处理器，来对一个或者多个工作流系统进行任务管理，例如提供公共的与用户交互的话框样式，独立于工作流管理软件中其他对话框所采用的样式。这可以联合几个工作流执行服务器，为终端用户提供一个单一的服务。② 支持两个由桌面环境控制的工作流执行服务器间的简单交互。

协同工作能力 —— 以支持**WAPI**调用和交换格式的接口**2**为基础。

一致性级别 —— 支持多种不同成熟度的可户端应用程序。

4.4.3. 应用程序和工具集成

目的 —— ① 使应用程序或工作能以一种标准的形式被工作使用（例如，通过活动控制功能与工作流机交互）；② 区分标准的应用程序与相似样式的非工作流可以使用的（**non workflow-enabled**）程序。

协同工作能力 —— 以支持**WAPI**调用来处理应用程序调用和工作流相关数据访问为基础。

一致性级别 —— 根据应用程序类型来划分。

4.4.4. 工作流执行服务器协同工作

目的 —— ① 支持使用不同的工作流执行软件产品，来开发自动化处理应用程序；② 使已有的工作流应用程序间进行应用程序或工作流相关数据的交换，这在两个过程中更经常发生。

协同工作能力 —— 以支持，使用活动网关或直接接口调用的**WAPI**调用和交换格式，为基础。网关模式被更多的应用程序采用；直接协同工作需要公共的过程定义交换和兼容的协议来支持。

一致性级别 —— 与协议支持，已在**3.7**中讨论过。

4.4.5. 公共工作流管理

目的 —— 支持几个工作流管理产品间的公共管理、审查功能

协同工作能力 —— 以支持**WAPI**调用的接口**5**为基础，通过公共的管理应用程序支持管理和监视功能。

一致性级别和使用协议—— 需要定义。

