# 提高分布式工作流管理系统的可扩展性

陶 冶　范玉顺　罗海滨
清华大学自动化系，北京 100084

**摘　要**　工作流管理系统为企业中不同部门之间的协作以及不同应用之间的集成提供了强有力的支持。在当前工作流技术的研究领域中，分布式工作流系统逐渐成为研究的热点，而系统的可扩展性是分布式工作流系统成功实施的一个重要因素。本文介绍了相关的领域概念、系统结构、过程实例的执行以及全局控制机制，提出了一种基于企业功能领域划分的、具有高度可扩展性的分布式工作流系统的设计和实施方案。

**关键字**　工作流；分布式工作流系统；可扩展性；柔性；CORBA

## 一、 引言

工作流技术作为现代企业实现过程管理与过程控制的一项关键技术，为企业的经营过程提供了一个从模型建立、静态分析和动态分析、仿真、实例化到运行的环境。同时，工作流管理系统通过一套集成化、可互操作的软件工具为这个框架提供了全过程的支持。

但是，也必须看到，工作流技术无论在理论上还是在技术本身还不够成熟[1]：工作流管理系统的实际应用还不是十分广泛，还不能全方位地支持企业的关键业务流程；各工作流系统间彼此几乎都不兼容，无法胜任大规模业务；系统可扩展性、可靠性和灵活性差，而且不能提供强有力的安全保证，这主要是因为单一数据库、集中式结构、有限的通讯能力以及缺乏前瞻性的设计等原因所致。

工作流管理系统从集中式演变成分布式是一个很大的进步，它使系统的整体性能大大增强。多工作流机的协作使系统从结构上变得更加合理，更加符合实际情况。可以说工作流机的分布是解决工作流管理系统可扩展性的关键。本文下面将介绍一种基于企业功能领域划分的分布式工作流管理系统的运行和管理模式。

## 二、 企业的功能领域

在企业 CIMS 应用工程的实施过程中，需要首先根据企业自身特点，利用通用功能视图模型的基本构件生成对企业需求分析阶段的功能模型。在 CIMOSA 体系结构中[4]，功能视图反映了企业的功能结构，从上到下，依次为领域（Domain）、领域过程（Domain Process）、经营过程（Business Process）、企业活动（Enterprise Activity）和原子级的功能操作（Function Operation），其中的领域是一个很重要的概念。根据企业的经营目标，企业可以划分为若干个互相正交的领域，每个领域都有其目标、功能及其领域过程，领域的功能由领域过程实现。领域的划分和确定由企业经营目标和所受到的限制条件共同决定。

根据过程视图与功能视图之间的交互特性以及企业功能领域划分的思想，我们将把工作流管理系统中的核心组件工作流机与企业的建模过程有机结合，使运行时的系统能够与企业模型紧密集成。这主要通过下面的设置来实现：首先，在建立过程模型时，通过领域配置功能将最基本的活动单元与适当的功能领域相关联，功能领域的信息由企业的功能模型提供；其次，对于每一个工作流机也同样为它们配置不同的功能领域，这可以在注册新的工作流机时完成设置。不同领域的工作流机负责执行不同的活动，这样使企业中的每一个任务与相应的工作流机对应起来，每一个工作流机的职责也随之变得清晰和明确。

## 三、 分布式工作流机的体系结构

按照 WfMC 提出的工作流参考模型[2]，一个工作流系统包括过程定义工具、工作流机、工作流管理工具、工作流客户应用和工作流机直接调用的应用等功能模块。

由于在实际应用中，工作流系统往往在不同的硬件平台、操作系统、网络协议、数据库的异构环境下运行，这对参考模型中各模块间的通讯和协作提出了很高的要求，CORBA 规范使这些要求可以较好的实现，为系统运行提供了一个功能强大的软件平台。

基于上面的考虑，本文提出下面的系统体系结构：

整个系统是由过程建模工具、模型仿真工具、一个总控工作流机、一个工作表管理器和多个执行工作流机组成，其中的工作表管理器实际上是一个功能固定的执行工作流机。

就执行机、工作表管理器和总控机的关系来说，它们构成了一个 CORBA 平台上的圆锥形模型，总控机位于锥顶，各执行机位于锥底的圆周上，工作表管理器位于锥底的圆心上。总控机能监控工作表管理器和任何一个执行机，工作表管理器和执行机向总控机报告自己的状态，工作表管理器记录和维护各执行机所执行的任务。从系统级别上说，各执行机彼此之间的地位完全对等，任何两个执行机之间都可以建立联系。它们之间的关系可以形象地用图 1 来表示：

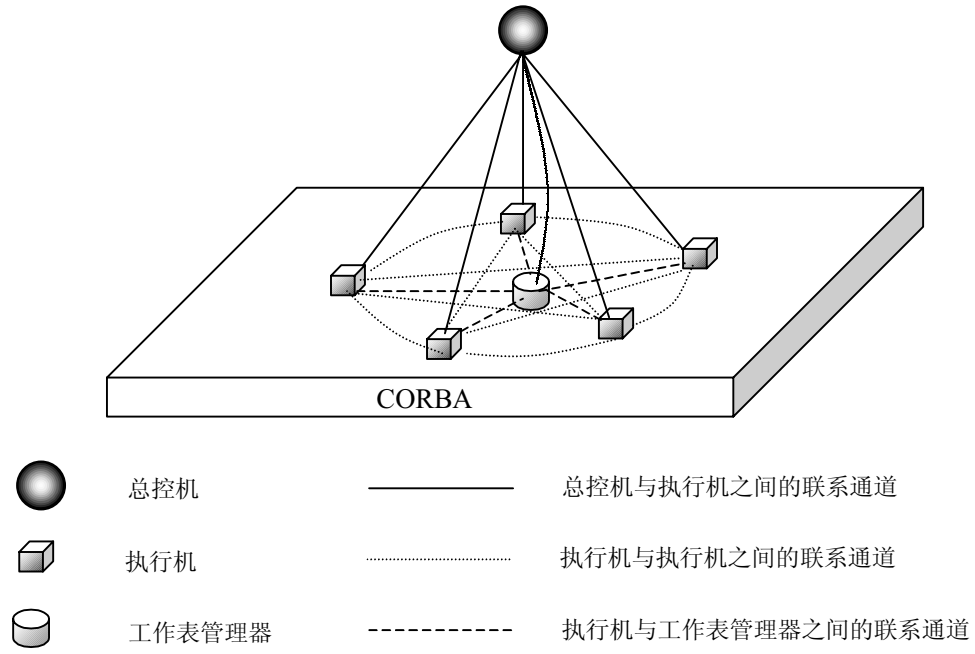


图 1　系统体系结构模型

建模工具负责创建工作流模型。模型中各个活动都应配置相应的功能领域，同时，各活动的输入输出、所需资源、活动间转移的条件、非相邻活动间的数据连接都需进行定义。模型建立后，在正式投入运行之前，须经过模型仿真工具的静态仿真和动态仿真，以检验其可行性并预估其运行效果。

总控工作流机与存储过程模型的中央数据库位于同一台机器上，主要负责指导建模工具进行模型分片、维护系统中各执行工作流机的配置信息、创建并激活过程实例、监控实例的运行状态等。执行工作流机主要负责生成活动实例、激活应用、在活动之间进行导航、与工作表管理器交互等。工作表管理器主要负责人工型活动任务项记录的生成、任务状态的更新与维护，这需要用户的参与。

整个系统采用基于 CORBA 规范的 Orbix Daemon 和 Orbix for Web 软件总线及其服务为系统支撑环境，系统提供的所有服务都经过 IDL 的定义与封装，经编译后以 CORBA 对

象的形式出现。这些对象遵循 IIOP 协议，能够接受嵌于 Web 的客户端的调用请求，正因如此，系统中所有的用户界面、管理员界面都是基于 Web 的。图 2 给出了系统的体系结构，并演示了一个简单的运行流程，其中的过程模型为简单的三个活动串行。

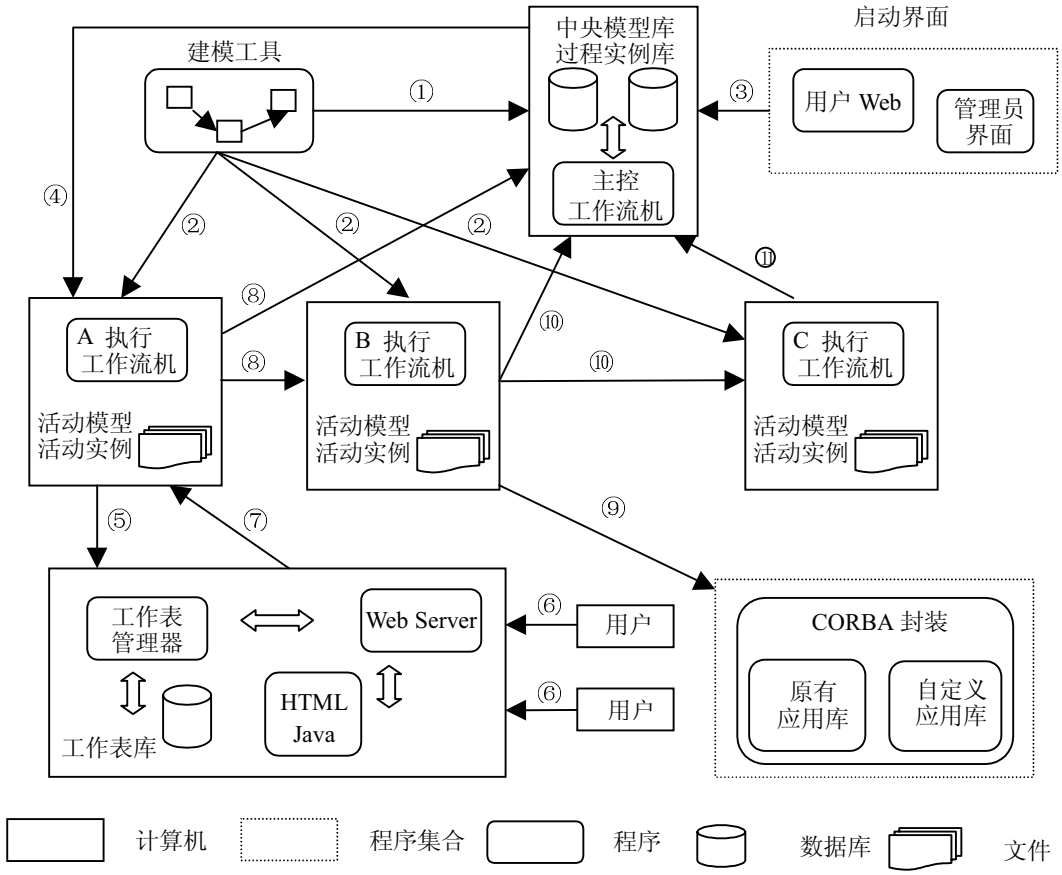


图 2　系统体系结构和实例运行流程

具体的运行步骤解释如下：

（1）通过建模工具，一个由三个活动组成的过程模型被建立，并被保存到中央数据库中；

（2）过程模型被分片传递到三个领域的工作流机上；

（3）该过程被某一启动界面启动，首先由总控工作流机生成一个过程实例，然后通过查找数据库中的模型信息及有关工作流机的配置信息，获得负责执行第一个活动的工作流机的主机地址；

（4）调用工作流机 A 的接口函数，由 A 生成新的活动实例；

（5）该活动是一个人工型活动，因此，A 将激活工作表管理器在数据库中生成相应的工作项，工作项的分配方式是采取用户自主的 PULL 类型；

（6）用户访问自己的工作表，并执行任务；任务完成后，用户通过同样的界面进行提交，系统更新工作表；

（7）工作表管理器通知执行工作流机 A，该活动实例已经完成；

（8）工作流机 A 通过本地的模型分片文件及配置信息，通知工作流机 B 执行过程实例中的第二个活动，同时通知主控工作流机它已经完成第一个活动；

（9）执行工作流机 B 调用其他应用程序来完成第二个活动，这是一个 CORBA 封装的自动应用；

（10）类似地，工作流机 B 通知工作流机 C 来执行第三个活动，同时通知主控工作流机它

已经完成第二个活动；

（11）C 通知主控工作流机，该过程实例已经执行完毕。

工作流运行环境的建立通常是先启动总控工作流机，然后启动工作表管理器和各个执行工作流机。系统运行时各执行工作流机均处于待命状态。企业可以根据自身的情况，按照实际要求来配置执行工作流机，比如每个领域或者每个部门配置一个。

图 2 中 CORBA 封装的应用包括系统自带的原有应用库和用户自定义的应用库，这些应用是一些诸如打印文档、发送邮件、定时、计数等功能单一且固定，从而可重用度较大的应用。这给用户提供了一个开放的接口，用户可以按系统提供的规范接口编写或设定他们用得较多的一些特殊应用。由于这些应用以软件组件的形式出现，在以后可以对之进行扩充和修改，从而可以提高系统的可扩展性和可重用性，便于企业信息系统的不断改进和重组。

## 四、 提高系统的可扩展性

系统的可扩展性是评价系统整体性能的一个重要指标，因为实际应用中生产任务或业务内容会随时发生变化，企业由于生产规模的变化，可能要对现在运行的分布式工作流系统进行不断的改组和扩充。工作流系统可扩展性好意味着新的功能可以很容易地加入到系统中，可以很容易地处理系统过载的情况而不会出现系统瘫痪或崩溃。

系统的可扩展性分为系统结构的可扩展性和系统功能的可扩展性。所谓结构的可扩展性是指整个系统运行的硬件平台能够根据需要很方便地进行规模上的扩充和压缩，而功能的可扩展性是指系统所具备的功能或能提供的服务能够根据需要进行添加或删除。由于功能与结构是紧密的，功能不能脱离结构而存在，所以要想获得功能的可扩展性，必须先保证结构的可扩展性。

从系统分析的角度来说，每一个系统都包括一些必不可少的关键功能和确保这些关键功能正常运行的关键资源，这些功能和资源统称为系统的关键组成要素[5]。关键组成要素集合包含的节点元素越多，则系统各部分之间的结合越紧密，关联越复杂，从而使系统的扩展越困难，可以说系统的可扩展性随着系统复杂度的提高而降低。因此，提高系统的可扩展性首先要从设计上降低系统组成结构的复杂度，其次要考虑降低系统中各元素间的关联复杂度。

综合以上两点考虑，要提高分布式工作流系统的可扩展性，就应该从提高系统结构和功能这两方面的可扩展性入手，通过尽量降低各工作流机之间的耦合强度，尽可能使分布式工作流系统在运行时能进行系统结构和功能的扩展。本文提出了以下两种提高系统可扩展性的实现策略，用来保证这一目标很好的实现。

**a. 使执行机之间以及执行机与总控机之间能动态进行注册和注销**

在系统中，总控机、执行机以及执行机提供的服务都通过 Orbix 提供的 Orbix Server Manager 进行注册，同时各服务对象对外提供透明的 IDL 接口。

以下是总控机和执行机各自与注册和注销相关的 IDL 接口：

```
struct EngineProperty
{
    string ip;                      //工作流机所在计算机的 IP 地址
    string name;                    //工作流机的名称
    string type;                    //工作流机的类型
    string domain;                  //工作流机所属的领域
    string current_start_time;      //工作流机本次启动时间
```

```
        string last_shutdown_time;          //工作流机上次关闭时间
        unsigned short serial_no;           //工作流机序列号
    };
    typedef sequence<EngineProperty> EnginePropertyList;

    interface WfControlEngine               //总控机接口
    {
        //注册新执行机
        boolean RegisterNewEngine(in EngineProperty new_engine， out unsigned short
serial_no，out EnginePropertyList other_engines，out unsigned short howMany);
        //关闭执行机
        boolean ShutDownEngine(in unsigned short serial_no);
        //获取执行机信息
        boolean GetEngineInfo(out EnginePropertyList engines， out unsigned short
howMany);
        //总控机提供的其它服务
        ……
    };

    interface WfEXEEngine                   //执行机接口
    {
        //更新配置文件
        boolean UpdateConfig(in EngineProperty own_property，in boolean flag);
        //执行机提供的其它服务
        ……
    };
```

一旦系统启动后，总控机在整个系统中主要是负责维护整个系统的配置信息，它向各执行机提供注册新执行机（RegisterNewEngine）、关闭执行机（ShutDownEngine）、获取执行机信息（GetEngineInfo）这三个相关的服务。每添加一个新的执行机或执行机的每一次启动都需调用总控机的 RegisterNewEngine 服务，该服务向执行机返回其它已经注册且处于运行状态的执行机的配置信息。类似地，每关闭一个执行机，它将调用 ShutDownEngine 服务，总控机将删除对应的执行机的记录。依靠这种方式，总控机在系统运行期间将一直保存着所有处于运行状态的执行机的配置信息。

执行机作为服务端可向其它执行机提供更新配置文件（UpdateConfig）的服务。每注册启动或关闭一个执行机，比如执行机 A，则执行机 A 将调用其它所有处于运行状态的执行机的 UpdateConfig 服务，以将执行机 A 的各项配置信息提供给它们，并将这些信息以配置文件的形式保存在各执行机本地，这使得任何执行机在运行时都知道其它处于运行状态的执行机的信息。

图 3 是执行机 A 启动和关闭的场景图：

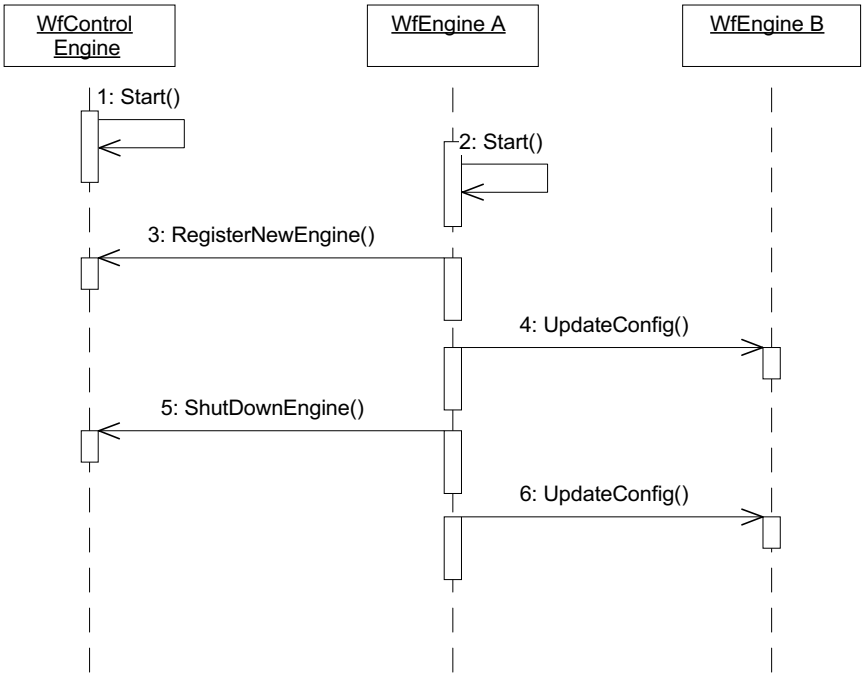


图 3　执行机启动和关闭场景图

从上面的 IDL 接口和执行机启动流程可以看出，各工作流机之间的关联仅在都处于运行状态的工作流机之间才存在，一旦某工作流机关闭之后，则它与整个系统的关联也就随之结束了。即使工作流机彼此之间存在关联，这些关联也只是每一个工作流机被动地接受其它工作流机的配置信息，除了总控工作流机和工作表管理器之外，各执行机之间的地位是完全平等的，不存在执行机彼此之间的控制、支配和制约。这样就避免了使某一工作流机成为系统的关键组成要素，降低了系统各组成要素之间的耦合强度，从而提高了系统的可扩展性。

**b.　将执行机与之提供的服务进行动态配置**

工作流机彼此之间的在线注册和注销仅仅从结构上提供了一个可扩展的机制。但如果每一个执行机所提供的服务都是预先设定好的，用户无法修改，则每向系统中添加一项新的服务都要添加新的执行机，这意味着系统功能的扩展必然导致系统规模的扩展和复杂度的增加，从而增加了整个系统的开销和监控管理的难度。因此，还应该从功能级上增强系统的可扩展性。将执行机与它所提供的服务进行分离，使服务能随时动态地配置到任何一个执行机上去，这是从功能上提高系统可扩展性的一个重要策略。

在系统中，每个执行机都配置在一台计算机上，因此这台计算机里就配置了一个可向建模工具、总控机、工作表管理器和用户提供服务的 CORBA 对象。同时，系统中的原有应用或用户自定义的应用，比如打印服务、传真服务、定时服务、电子邮件服务等等，都通过 CORBA 封装成一个个独立的服务实体。这些实体也通过 Orbix Server Manager 进行注册，动态配置到执行机所在的计算机里。从本质上看，虽然执行机和上面这些应用都是系统提供的服务对象，但实际上可以认为这些应用是外挂在执行机下的，因为在系统运行中导航到某台执行机后，用户就能获得这台执行机所能提供服务的一个清单。

有了上面这种策略，在系统内部就可以自由地实现服务的动态配置，这样当系统功能需要扩充时，只需根据一定的规则将相应的服务随时配置到某一个或几个执行机下，而无需添置新的计算机。这样既降低了系统的开销，也降低了对系统进行监控管理的难度，有效地提高了系统的可扩展性。

## 五、结论

本文所提出的分布式工作流机的方案，按照对企业功能模型的需求分析与设计说明，工作流管理系统在过程定义与执行这两个层次上对企业的提供支持。在定义过程时，将最基本的活动单元与适当的功能领域相关联；在执行过程时，不同的活动由不同领域的工作流机来执行。同时，系统采取了多种措施来提高系统的可扩展性。这种工作流机的分布式方案不仅可以解决原有集中式工作流管理系统所存在的一些瓶颈，而且能够与企业的建模过程紧密集成，从而形成一个完整的、具有良好可扩展性的、从建模到实施的过程体系，能较好地满足企业实际生产和业务流程的要求。

**参考文献**


1 陶冶，范玉顺，罗海滨，“分布式工作流系统的可靠性研究”，《计算机科学》，Vol

2 范玉顺，吴澄，“工作流管理技术研究与产品现状及发展趋势”， 《计算机集成制造系统 CIMS》，Vol.6，No.1，Jan.，2000，pp.1-7

3 Haibin Luo, Yushun Fan, “CIMFlow: A Workflow Management System Based on Integration Platform Environment”,7th IEEE International Conference on Emerging Technologies and Factory Automation,1999，pp.233-241

4 Alonso G, Agrawal D, Abbadi El A, et al. Functionality and Limitations of Current Workflow Management Systems. IEEE Expert, 1997,12(5).

5 王景光，甘仞初. “信息系统结构可扩展性分析”，System Engineering and Electronics，Vol 21，No.9，p37~39，1999

6 WfMC. Workflow Management Coalition Terminology and Glossary (WfMC-TC-1011). Technical Report, Workflow Management Coalition, Brussels, 1996.

7 William Golden, Philip Powell. Towards a definition of flexibility: in search of the Holy Grail? Oemga 28(2000) - The International Journal of Management Science. P373-384


# Improving the Scalability and Flexibility of a Distributed Workflow System


Tao Ye, Fan Yushun, Luo Haibin

Department of Automation，Tsinghua University，Beijing 100084



**Abstract** Workflow management system (WfMS) has been used to provide a powerful computation environment for the integration of different applications and the cooperation of different organizations within enterprises. In the current research field of workflow technologies, the distributed workflow engines have gradually caught more and more attentions, and the scalability and flexibility of the whole system are the two critical factors to the successful implementation of the distributed workflow system. In this paper, a distributed workflow system with high scalability and high flexibility that runs on different enterprise function domains is introduced. Meanwhile, the relevant concept of domain, the system infrastructure, the implementation of a process instance and the global control mechanism are also presented.

**Key words** Workflow；Distributed Workflow Engine；Scalability；Flexibility；CORBA