# 一种高性能的分布式工作流系统实现框架

张志君，范玉顺
（清华大学自动化系，北京 100084）

**摘要：**指出现有工作流系统的缺点和企业的实际需求，提出了一种基于 COM+技术的分布式工作流系统实现框架，通过多层结构的方式实现了工作流系统中应用、逻辑与数据的分离。进一步分析了系统在可靠性、可扩展性与实用性等方面的性能，并给出了相应的实现策略。实际应用初步证明了基于该框架实现的工作流系统适用于企业业务过程管理。

**关键词：**工作流；分布式工作流系统；可靠性；可扩展性；实用性；COM+
**中图分类号：**TP319

## 1 引言

工作流技术是实现企业业务过程管理与控制的一项关键性技术。目前，工作流技术已经成为了计算机应用的一个重要研究方向，并被逐步应用于服务业与制造业等领域[1]。但是从工作流系统的实际应用状况来看，还远未达到人们所期待的普及程度。限制当前工作流技术广泛应用的主要因素之一就是现有工作流系统在可靠性、可扩展性以及实用性等方面都无法全面满足企业实际运行环境的要求。

在当前工作流技术的研究领域中，分布式工作流系统已逐步成为研究的热点[1-3]。现有的分布式工作流系统确实可以在一定程度上解决集中式系统所固有的性能瓶颈、单点失败与可扩展性差等问题。但通过对基于持久消息队列的 Exotica、基于事件的 EVE、基于可移动代理的 DartFlow 等分布式工作流系统的分析[1,2]，我们发现这些系统在实用性方面仍存在以下问题：完全分布式的结构增加了系统应用和维护的复杂性；工作流系统中数据一致性维护困难，尤其是在涉及到模型动态修改和异常处理时问题尤为突出；将执行期的分布方式定义到工作流模型中，降低了系统的柔性；系统分布策略仍属于静态负载均衡。实际情况却是，企业更希望工作流系统能够提供集中可靠的数据管理（尤其是分布式数据库技术发展并不理想的情况下）、动态的系统负载均衡以及支持部门级别或企业级别的多工作流系统协作功能。

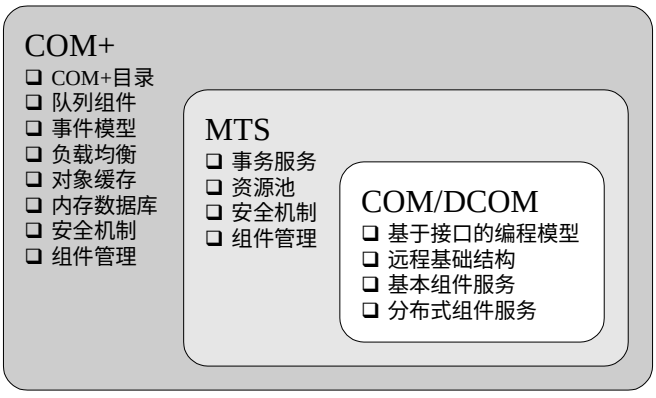


图 1 COM+技术的功能扩展

收稿日期：2002-9-15
基金项目：国家自然科学基金资助项目（60274046），国家 863 计划资助项目（2002AA414710）。Foundation item: Project supported by National Natural Science Foundation of China (No. 60274046) and State High-Technology Development Program of China (No. 2002AA414710).
作者简介：张志君（1978.10-），男，汉族，清华大学自动化系硕士研究生，研究领域为 CIMS 环境下面向企业的工作流技术。Email: zzj00@mails.tsinghua.edu.cn.

COM（Component Object Model）是微软开发的一项具有位置透明性、语言无关性、版本升级鲁棒性等特征的系统级别的面向对象组件技术，它通过接口方式定义并实现不同软件模块之间的通讯与协作功能[4]。COM+在 COM、DCOM（Distributed COM）和 MTS（Microsoft Transaction Server）基础上进一步优化了组件管理环境与事务服务，并提供队列组件、事件模型、负载均衡、对象缓存、内存数据库以及安全机制等服务内容扩展，如图 1 所示。COM+把 COM 模型推向了更高层次，使得其成为了企业级别分布式软件开发领域的重要技术。

基于上述考虑，本文从工作流应用的实际需求出发，提出了一种基于 COM+多层结构方式的分布式工作流系统实现框架，并分析了按照该框架结构实现的工作流系统在可靠性、可扩展性与实用性等方面的优势，同时结合工作流系统的实现给出了一系列增强系统性能的策略。

## 2 分布式工作流系统的实现框架

按照工作流管理联盟提出的工作流参考模型，工作流系统应包括过程定义工具、工作流机、工作流管理工具、工作流客户应用和工作流直接调用的应用等功能模块[5]。在该参考模型的基础上，我们提出了基于 COM+技术的多级分布式工作流系统实现框架，基本可以划分为应用层、逻辑层和数据层三个层次，如图 2 所示。

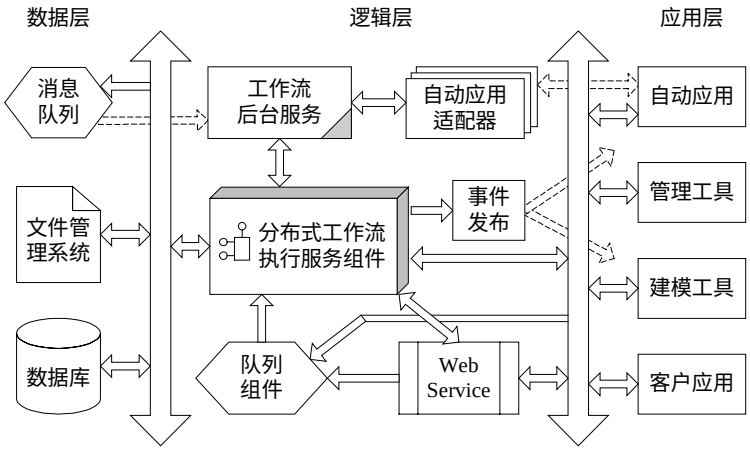


图 2 基于 COM+技术的分布式工作流系统实现框架

### 2.1 数据层

数据层是工作流系统的数据基础，由数据库、文件管理系统、消息队列三个服务器组成。其中数据库保存工作流系统中的模型数据、实例数据和日志记录，文件系统辅助数据库保存文件类型数据，消息队列则用来存储和管理工作流系统中的业务事件信息。在工作流系统中引入业务事件的概念使得在过程实例内部或不同的过程实例间，甚至不同工作流系统间可以通过业务事件方式进行交互。

### 2.2 逻辑层

逻辑层是工作流系统功能实现的核心部分，主要包括分布式工作流执行服务组件、工作流后台服务、自动应用适配器、队列组件、Web Service 和事件发布六个基本模块。下面主要介绍工作流执行服务组件和工作流后台服务两部分内容，其他模块将在后文详细分析。

分布式工作流执行服务组件是该层的核心组成模块，它通过将工作流相关业务逻辑封装为 COM+服务器的方式，为逻辑层其他模块和应用层提供访问和操纵数据层的标准接口。由于工作流执行服务组件是以无状态 COM+组件方式实现的，所以该组件内部不会也不需要持久保存任何工作流数据，所有工作流系统状态信息都将保存在数据层的服务器中。分布式工作流执行服务组件实现如下的接口定义，其中包含了九个主要的功能实现：

```
interface IWfES : IDispatch
{
    // Part I. Connection Functions
    [id(1)] HRESULT Connect ([in]BSTR Server, [in]BSTR User, [in]BSTR Password,
[out, retval]long* UserID);
    [id(2)] HRESULT Disconnect ();

    // Part II. Process Instance Functions
    [id(3)] HRESULT CreateProcessInstance ([in]BSTR ProcDefID, [in]BSTR
ProcInstName, [out, retval]long* ProcInstID);
    …
    // Part III. Activity Instance Functions,  // Part IV. Relevant Data Functions
    // Part V. Work Item Functions,            // Part VI. Process Definition Functions
    // Part VII. Log Functions,                // Part VIII. Business Event Functions
    // Part IX. File Operation Functions
    …
};
```

工作流后台服务是工作流系统在执行服务接口基础上提供的一套功能性辅助服务，主要负责实现时间管理、条件管理、自动应用管理、组织资源分配等功能。时间管理负责执行用户定义的计划任务和检查工作流执行过程中的时间约束；条件管理负责执行等待条件的判断，包括数据表达式条件、时间类型条件和事件类型条件三种，其中事件类型条件通过查询消息队列服务器的业务事件信息进行处理；自动应用管理负责激活和管理自动应用程序；组织资源分配为已经创建的活动分配必要的执行者和资源。由于这些功能在一定程度上都具有时延或时耗特性，而以 COM+服务器方式实现的分布式工作流执行服务组件并不适合于实现这类功能，这正是增加工作流后台服务处理这类业务逻辑的主要原因。后台服务的运行是相互独立的，可以同时启动多个后台服务，并且各自配置不同的实现功能。另外，对于组织资源分配功能，系统定义相应的接口，支持动态装载组件。这样如果工作流系统提供的缺省实现方式不能满足用户的特殊要求，用户可以自定义实现组件，然后由系统自动装载运行，甚至可以对于不同的过程使用不同的组织资源分配组件。

```
interface IWfPRAssign : IDispatch
{
    [id(1)] HRESULT AssignPerformer ([in]long ProcInstID, [in]long ActInstID,
[out]BSTR* xmlPerformer, [out]long* ErrorCode);
    [id(2)] HRESULT AssignResource ([in]long ProcInstID, [in]long ActInstID,
[out]BSTR* xmlResource, [out]long* ErrorCode);
    …
};
```

### 2.3 应用层

应用层由建模工具、客户应用、自动应用和管理工具组成。其中客户应用和自动应用还包含在系统实施过程中针对企业具体业务内容所定制开发的应用，这些应用都将运行在工作流系统的集成环境中。应用层对工作流系统数据的操纵，都是通过工作流执行服务组件提供的接口实现的。

显然，在 COM+技术提供了强大的应用分布与底层通讯基础上，三层结构很好的实现了工作流系统中应用、逻辑与数据的分离。应用与逻辑的分离规范了应用对数据的操作，

增强了系统的可重用性，使得应用开发的重点放在与用户的交互之上；逻辑与数据的分离优化了数据的共享机制，增强了数据安全性。

## 3 系统性能分析与提高

### 3.1 可靠性

**1) 工作流数据的集中、持久存储**

工作流数据在系统中是集中、持久存储的，这种存储方式使得数据的一致性得以保证，并为数据的管理与维护带来很大方便。而且，即使在工作流数据服务器无法正常工作的情况下仍可以方便的通过日志进行错误恢复，这样就使得系统的异常处理能力和错误恢复能力得到了相当程度的提升，系统具有很好的可靠性。

当然，数据的集中存储也会给系统性能带来一定的负面影响，数据服务器的处理效率与网络带宽可能会成为系统的瓶颈。采用分布式数据库技术可以在一定程度上解决数据服务器瓶颈的问题，但是分布式数据库技术目前还处于研究阶段，应用方法非常复杂而且不成熟，市场上也缺乏真正支持完全分布的数据库产品。鉴于上述原因，我们通过（1）数据访问采用存储过程；（2）合理化执行服务器的负载配置；（3）使用内存数据库三种方法平缓和减少网络流量，提高数据处理效率，从而降低数据服务器成为系统瓶颈的可能性。

**2) 基于事务方式的数据协调访问**

数据层还为整个工作流系统提供了强大的数据协调访问功能，不同工作流执行服务组件通过数据库来实现工作流数据的共享。系统运行过程中，按照具体业务逻辑动态的选取不同的数据库事务隔离等级（包括未提交读、已提交读、可重复读和串行化四个等级）可以非常有效的解决多工作流执行服务组件并发访问数据所导致的访问冲突问题，从而实现工作流数据的实时高效共享。由于工作流执行服务组件是以无状态COM+组件方式实现的，组件内部不会持久保存任何工作流数据，所以该组件对工作流数据的访问都是由某一外部事件所激活，并且以事务方式进行，这对于保证数据的完整性与可靠性也相当重要。

**3) 良好的日志记录与数据备份**

在企业这样一个复杂的应用系统中，出现异常和错误是相当常见的情况。良好的系统日志机制是保护系统数据一致性，提高系统异常处理能力与错误恢复能力的重要手段。系统通过日志方式记录下每一事件发生所产生的结果，这样就可以在必要的时候根据记录进行回滚与恢复。而且日志记录还为企业使用工作流执行数据进行数据挖掘提供了详细的历史资料。另外用户还可以通过添加数据服务器备份机或对数据库进行定期备份的方式，将工作流历史数据备份到其他介质中，防止由于数据系统崩溃而引起的工作流数据丢失。

### 3.2 可扩展性

一般说来，工作流系统的可扩展性包括两个方面：功能可扩展性和结构可扩展性。功能可扩展性主要是指系统在现有功能基础上扩展新的功能与服务的能力，良好的系统集成性和开放性是系统功能可扩展性的重要体现；而结构可扩展性主要指系统在复杂的网络环境中通过结构的拓展适应系统规模或业务容量变化的能力，松散耦合的系统结构和良好的分布式机制是系统结构可扩展性的重要体现。当然系统的功能可扩展性与结构可扩展性之间是紧密关联的，提高系统整体的可扩展性就需要对这两方面综合考虑。

**1) 接口方式的自动应用集成**

集成性是增强工作流系统功能可扩展性的一个重要方面。一般说来，工作流系统集成自动应用可以采用直接激活、消息传递、协议通讯和接口封装等方式。其中接口封装是最为灵活的一种。自动应用实现工作流系统规定的接口就能被系统集成，对于无法直接实现该接口的应用还可以通过提供自动应用适配器的方式实现。通过接口方式的自动应用集成，工作流系统不仅可以将企业现有应用集成进来，同时也可以将相当一部分的工作流系统功

能通过自动应用的方式来实现，而不是硬编码到系统的原始实现中。这种方式对于专用的、选择性的工作流功能将更为灵活。下面给出了自动应用集成的 COM+接口，具体实现时，可以给出对于常见的 EJB 组件、CORBA 组件、普通 EXE 和 DLL 的自动应用适配器。

```
interface IWfAutoApplication : IDispatch
{
    [id(1)] HRESULT Run ([in]BSTR xmlRelvDataList, [out]BSTR* xmlOutRelvDataList,
[out]BSTR* AbortReason, [out]long* ErrorCode);
    [id(2)] HRESULT Terminate ();
    [id(3)] HRESULT GetStatus ([out]long* Status);
    …
};
```

**2) 开放式的事件发布**

作为企业业务过程运行平台的工作流系统除了要集成企业现有应用外，还应该提供一种供外部应用监听工作流系统运行情况的开放式接口。事件发布以 COM+事件的方式将工作流系统中的事件信息向外发布，这样应用就可以通过订阅事件并在事件发生时获得通知。工作流系统发布的事件可以包括工作流系统内部事件和业务事件，事件发布与否应该在工作流模型中定义。由于 COM+事件是一种松散耦合的事件服务，所以企业可以通过这种开放式的接口进一步开发新的扩展应用，增强工作流系统的功能可扩展性。

**3) 队列组件与 Web Service 封装**

队列组件和 Web Service 都是为了降低了应用层与工作流执行服务组件之间的耦合程度，提高系统结构可扩展性而提供的工作流执行服务组件访问封装形式。

队列组件为用户提供基于消息传递方式的工作流执行服务接口调用。一般来说，COM+服务以 RPC 调用方式实现，应用程序对组件接口方法的调用是同步的，应用程序的生存期和组件实例也是紧密耦合的。在工作流系统中，这就要求应用程序必须保证能够连接到工作流执行服务器，并一直等待接口调用的返回，显然这无法满足诸如离线操作等应用的要求。队列组件将应用对工作流执行服务接口的调用分为两个步骤，首先是应用对队列组件的调用，然后才是队列组件对工作流执行服务接口的调用，从而实现了应用程序与工作流执行服务组件之间的消息传递，降低了二者的耦合程度。队列组件解决了普通 COM+服务中客户与组件之间的同步限制，尤其使得客户程序的离线工作成为可能，同时队列组件对于平缓工作流系统负载也是一种有效的途径。

Web Service 则为应用程序通过 Internet 访问工作流执行服务接口提供了方便。这种基于 SOAP（Simple Object Access Protocol）协议的访问从根本上解决了 COM+技术跨越 Internet 防火墙时所存在的问题，使得整个工作流系统在 Internet 上分布成为可能。另外，Web Service 可以直接对工作流执行服务接口进行封装，也可以进一步封装队列组件接口，从而使得工作流系统在 Intranet 中所提供的功能能够完全应用于 Internet 中。特别是涉及到企业各部门间以及不同企业间工作流系统的协调处理时，这种 Web Service 形式的封装就更显得必要而且有效。

**4) 基于动态负载均衡服务的工作流系统部署**

负载均衡服务可以按照给定的规则在一组计算资源上进行任务分配，以期有效的实现企业现有资源的充分利用，解决企业级别分布式计算的可扩展性与灵活性等问题。CLB（Component Load Balancing）是微软为实现 COM+组件动态负载均衡功能而提供的一项负载均衡技术[6]。如图 3 所示，在工作流系统中采用 CLB 技术时，系统包括数据服务器（可细分为数据库服务器、文件服务器和消息队列服务器）、负载均衡路由器、由多个执行服务器组成的执行服务器簇以及各种类型的客户机。工作流执行服务组件、工作流后台服务和

自动应用程序将被部署在各台执行服务器上。在负载均衡路由器收到客户机的工作流执行服务组件创建请求后，它通过负载均衡引擎确定执行服务器簇中负荷最小的机器，并向其发送创建请求，一旦对象创建后，该执行服务器和客户程序之间就存在了直接连接。默认的 COM+负载均衡引擎是通过响应时间算法来确定由哪台服务器响应请求的。响应时间算法就是通过将给定机器上每一实例的每一接口的每一方法的调用时间合计起来，以此动态确定该机器的负载情况，当然也可以根据具体情况编写针对性的负载均衡引擎。另外，COM+负载均衡服务对于用户来说是透明的，所以客户只需要连接到负载均衡路由器即可，而无需关心具体针对的是哪台执行服务器。工作流系统的客户机包括普通客户机、远程客户机、Web 服务器、其他内部的工作流系统，甚至远程工作流系统。多工作流系统间的相互协作可以建立在业务事件或者过程激活与同步的级别上，满足企业内部部门间和企业间业务的松散耦合要求。负载均衡服务的引入使得系统的动态性能和系统的可扩展性都得到了很大程度的提高。

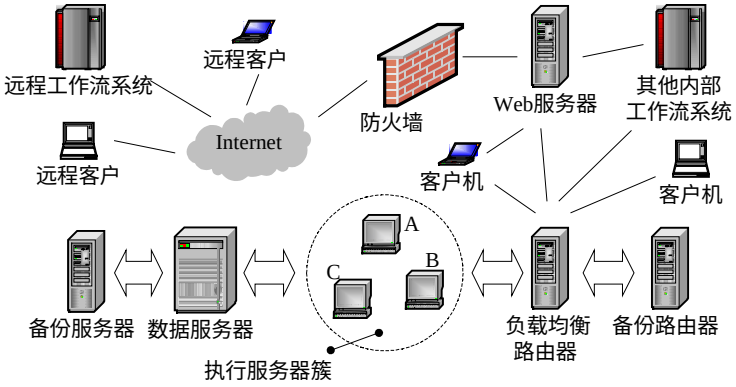


图 3 采用 CLB 的工作流系统分布式部署方案

### 3.3 实用性

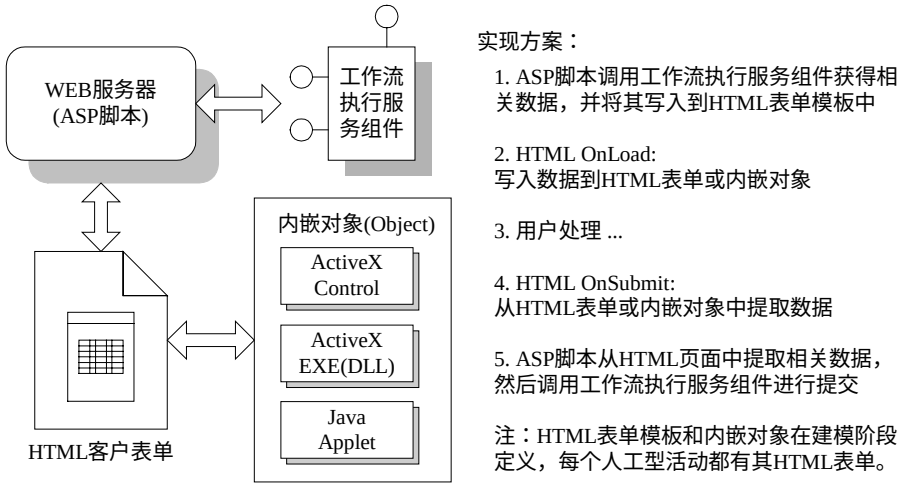


图 4 基于 Web 的客户端实现方案

在实际的企业业务运行过程中，除了要求工作流系统具有良好的性能外，方便实用也是企业选取工作流系统时所考虑的一个重要因素。与 CORBA、J2EE 等其它技术相比，采用 COM+技术实现的工作流系统具有易实施，易维护，低成本的优点，而且可以方便的与 Web 技术相结合，尤其是 COM+组件的许多配置操作都与操作系统直接集成，这都为企业用户的直接使用和二次开发提供了极大的方便。举例来说，如图 4 所示，在基于 Web 的客户端实现方案中，系统可以提供 HTML 形式的用户表单供用户处理工作项，还可以将

ActiveX Control、ActiveX EXE(DLL)、Java Applet 等客户应用集成进来，协助用户完成工作项。

## 4 结论

本文提出的分布式工作流系统设计框架，以 COM+技术为应用分布和底层通讯基础，通过三层结构的方式很好的实现了工作流系统中应用、逻辑与数据的分离，使得系统在可扩展性、可靠性与实用性方面都具有良好的性能。目前，以此为原型的分布式工作流管理系统 CIMFlow 3.0 已经基本实现，测试和在生产管理系统、ERP 系统以及电子政务中的初步应用都表明该系统能够满足企业在集中可靠的数据管理、动态的负载均衡以及支持多工作流系统间的协作功能等方面的要求，与企业现有应用结合形成一个完整的过程体系。同时本文所给出的系统功能实现策略对于提升实际运行环境中系统的动态性能也是相当有益的。

## 参考文献：

[1] FAN Yu-shun, LUO Hai-bin, LIN Hui-ping, et al. Fundamentals of Workflow Management Technology[M]. Beijing: Tsinghua University Press, 2001. (in Chinese) [范玉顺, 罗海滨, 林慧萍, 等. 工作流管理技术基础[M]. 北京: 清华大学出版社, 2001.]

[2] Alonso G, Mohan C, Gunthor R, et al. Exotica/FMQM: a persistent message-based architecture for distributed workflow management[A]. In Proc. of IFIP WG 8.1 Workgroup Conference on Information Systems Development for Decentralized Organizations[C]. Norway: 1995.

[3] TAO Ye, FAN Yu-shun, LUO Hai-bin. Improving the Scalability of Distributed Workflow System. Computer Sciences[J], 2001, 28(6): p.30-33. (in Chinese) [陶冶, 范玉顺, 罗海滨. 提高分布式工作流管理系统的可扩展性[J]. 计算机科学, 2001, 28(6): 30-33.]

[4] Tom Armstrong, Ron Patton. ATL Developer's Guide[M]. IDG Books Worldwide Inc, 2000.

[5] Workflow Management Coalition. The Workflow Reference Model (TC00-1006)[S]. Workflow Management Coalition, 1995.

[6] Microsoft Corporation. Microsoft Application Center 2000 Component Load Balancing Technology Overview[R]. Microsoft Corporation, 2000.

**A Realizing Architecture with High Performance for Distributed Workflow System**

ZHANG Zhi-jun, FAN Yu-shun

Department of Automation, Tsinghua University, Beijing 100084, China

**Abstract:** Current workflow systems have drawbacks in their practicability and cannot meet the actual requirements of contemporary enterprises. In this paper, a realizing architecture of COM+ based distributed workflow system which can achieve the isolation of application, logic and data by a N-tier framework is represented. Then, the performances in reliability, scalability and practicability of the workflow system based on this architecture are further investigated, and also some implementing methods are discussed. Preliminary applications show that workflow systems based on this realizing architecture are applicable for the managements of enterprise business processes.

**Key words:** Workflow; Distributed Workflow System; Reliability; Scalability; Practicability; COM+