实体驱动业务的企业信息系统软件框架*)

杨喜敏[1,2] 谢长生[1]

(华中科技大学计算机科学与技术学院 信息存储系统国家专业实验室暨教育部重点实验室 武汉 430074)[1]

(中南民族大学计算机科学学院 武汉 430073)[2]

摘 要 现代企业日益强化企业信息系统的开放性、可重用性和可伸缩性。本文针对当前企业信息系统研发中的一些问题,在将企业信息系统可被抽象表示为"用户在安全控制下对受控实体的一组业务集合的调度"的基础上,提出了实体驱动业务的信息系统软件框架 EDTISF。重点讨论了 EDTISF 如何以"受控实体"来实现企业业务的确切划分,如何通过独立的数据对象来降低业务及其处理数据的耦合度,以及如何在应用层的级别上实现业务调度机制的重用等问题。

关键词 企业信息系统,受控实体,实体驱动业务,信息系统框架

Entity-driven Task Software Framework for EIS

YANG Xi-Min XIE Chang-Sheng

(Department of Computer Science, Huazhong University of Science & Technology, Wuhan 430074)

Abstract In this paper, we present that EIS (Enterprise information systems) can be abstracted to a scenario in which users could schedule the task suites of controlled entities under enterprise security mechanism, and propose a novel Entity Driven Task Software Framework (EDTSF) for EIS. Based on existing hierarchical information system architectures, the EDTSF could clearly implements enterprise business partition, reduces coupling between data objects and tasks; it also increases flexibility and expansibility of the EIS by expanding service function of data object and reusing the scheduling mechanism at system level. We have noticed that little research has been done on the partition rules and methods of enterprise business. The EDTSF is a new software framework and its application results show that the EDTSF is an effective approach to analyzing, designing and implementing EIS.

Keywords Enterprise information system, Controlled entity, Entity-driven task, Information system framework

1 引言

企业信息系统 EIS(Enterprise Information System)已经成为现代企业中不可或缺的基础设施之一。企业业务的持续性变化和信息资源全面数字化的需要、EIS 网络化和集成化的趋势,使得现代企业日益强化信息系统的开放性、可重用性和可伸缩性[1],要求 EIS 必须具有高度的敏捷性,能够依企业的需求变化而即时调整和重构。

近年来,国内外学者在企业体系结构 EA(Enterprise Architecture)和信息系统体系结构 ISA(Information System Architecture)等方面取得了诸多研究成果。既有 SOA(Service Oriented Architec- ture)[5,6]、CBSA(Component Based Software Archi- tecture)[2,4]、SF(Software Framework)[4]等软件工程方法在 EIS 中的应用研究,也出现了 ZA(Zachman Architecture)[7] 和 AOIS (Agent-Oriented Information System)[8,9]等专用于 EIS 的 EA 和 ISA,以及这些技术的应用研究,如文[1,3,10~13]等。然而,这些研究成果的抽象层次决定了它们除了能够提供概念性的指导外,并不能直接向 EIS 研发人员(尤其是初级的)提供更多的实践性指导。其次,由于目标对象和侧重点的不同,许多技术还并不具有普遍适用的特性。例如:构件技术的方法论尚待进一步完善;SOA 则存在没有严格区分企业系统与企业服务、忽视了企业系统的动态性和不包括任何管理的含义等问题[12]。

目前可用于 EIS 构建中的支撑技术也很多,主要有:1)中间件平台技术,如 CORBA、J2EE、.Net 和 Web Service 等;2)用于企业过程集成的各种工作流技术和业务过程管理技术;3)用于保持 EIS 良好的重用性和开放性技术,如 MDA 和各种体系结构描述语言 ADL(Architecture Descrip- tion Languages)技术;4)用于动态地创建和管理各种企业应用服务的 GRID 技术;5)用于管理分布式环境下的 EIS 的 WBEM 技术等。但能够将它们综合应用于 EIS 的工作还很少[12]。

上述研究中普遍关注的问题是企业业务功能的实现和集成,以及高效的企业信息资源共享机制,而较少涉及对企业业务划分规则的研究,不能直观地体现企业资源的管理层次结构,缺乏有效的应对企业资源调整和重组的机制。传统的基于角色进行职责划分的方法[14]重点是研究企业信息安全和访问控制等问题,并不适宜解决这些问题。

层次化信息系统模型能够从不同层次上实现系统的可重构性,可即时实施软件调整以适应企业需求变化[15]。本文针对当前 EIS 开发的问题,在现有 EIS 层次模型的基础上,重点探讨了如何实现企业业务的确切划分和降低业务及其处理数据间的耦合度等内容,提出并讨论了实体驱动业务的信息系

*)本课题得到国家自然科学基金(60303031)、国家重点基础研究发展规划 973(2004CB318203)资助。杨喜敏 博士研究生,主要研究方向为企业信息系统、海量信息存储系统等;谢长生 教授,博士生导师,研究方向为网络存储和高密度光存储技术。

统框架 EDTISF(Entity Driven Task Information System Framework)及其关键技术等。与文[3,10～12]等重点强调业务构件和功能构件的重用,进而实现系统重构的方法不同,我们更注重的是如何在应用层上实现可重用的业务调度机制。

本文将在第2节中简单讨论当前EIS构建中的主要问题,在第3节中叙述EIS的抽象化表示和形式化定义,并对提出的EDTISF框架及其关键技术进行讨论,在第4节中给出EDTISF与其他EIS开发技术的比较分析;最后总结全文并给出进一步的工作展望。

2 问题及分析

EIS研发涉及的问题很多,包括EA、框架、组件辨识(Component Identification)等[20～25]。这里重点讨论和分析以下几个问题。

2.1 既有应用系统的广泛异构性

当今计算机应用领域的发展为企业信息系统设计带来了复杂异构环境间的互操作问题[12]。这也是企业应用集成EAI(Enterprise Applications Integration)一直是EIS领域研究热点的主要原因之一[2～4]。企业既有应用系统(尤其是在不同历史时期构建的部门级应用系统)通常是按固定的职能需求来设计和设施的,由于采用的技术不同(如开发语言、指导原则等)、软硬件平台的差异和数据交换机制的自主性等,导致系统间甚至是在同一系统的不同组件之间,都存在着广泛异构的特性。因此,EIS构建时的首要任务就是既有应用系统的移植和集成,在有效保护企业既有投资的同时,实现企业信息资源的最大化共享和充分利用[14]。

传统的EAI方法本质上是点对点的集成,容易受制于传统分布式对象中间件技术存在的局限性,如CORBA、DCOM、JavaRMI之间的互操作性、客户端与服务端之间的紧耦合等[13]。面向服务、面向代理和企业信息总线EIB(Enterprise Informa- tion Bus)等技术则能够摆脱这些局限性。SOA通过位置透明的服务组件实现了客户与服务之间或服务间的松散耦合,可以在不改变应用系统原有底层架构的基础上实现灵活的、面向服务的应用集成。不过,作为系统间的数据交换媒介,服务组件或代理对象在应对企业信息的动态性时功能负担过重。EIB借用计算机体系结构中的总线概念,强调部门间的信息流动,可提供不同系统数据传输的高速通道,同时实现数据基于标准的转换,有利于简化系统的复杂性,增强系统的可扩展性和兼容性。

2.2 服务、框架与构件技术的问题

SOA、SA和CBSA能提高EIS的开发质量和效率,目标系统企业业务流程变化的适应能力也较强。但是,目前基于构件或框架的开发方法还处于研究阶段,尤其是在如何快速、可靠地应用可复用构件来构建一个构件化的应用系统等方面,还有很多需要深入研究的内容[10,16]。而SOA在安全性、事务一致性和服务的管理等很多关键技术方面尚不成熟,也未形成统一的标准[12]。

其次,SOA、SA和CBSA的抽象层次决定了它们能够为EIS开发人员提供基本的概念性指导外,不能提供更多的实践性指导。开发人员必须自主填补这些技术开发时所假设的体系结构与实际应用间的差异。由于构件的功能表示及构件间交互机制和接口定义往往是不同的,因此构件隐蔽的细节越多,EIS中外来构件的数量越大,系统管理和维护人员在EIS运行和维护期碰到的问题就会越多。此外,如何实现应用领域问题与服务、框架或构件的实现映射等,也还是亟待解决的问题[2,16～19]。

2.3 对象的持久化问题

现行EIS数据存储和开发方法的事实性标准是关系数据库和面向对象技术。对象由数据和行为组成,而一个关系型数据库则是由表和它们之间的关系组成的。这些本质上的不同,导致EIS开发人员必须面对如何在关系数据库中实现对象的持久化等问题,包括对象的存储机制,以及如何为应用者提供透明的、健壮的应用方法等。

针对存储系统的复杂性和信息组织结构的易变性,常用的持久化对象表示方法是对象-关系映射。但由于关系数据库本质上缺乏表示持久化对象所需要的基础构件,易于造成对象-关系阻抗不匹配的问题。面向对象的数据库是由对象实际组成的,本身就支持对象的持久化表示。但是,面向对象数据库中,类定义的改变往往造成高昂的数据库模式迁移代价,而且其对即席查询(ad-hoc queries)的支持也远不及关系数据库[20]。其他的一些技术,如在类中直接嵌入SQL命令和特定的数据对象或是数据中间件等,则有可能导致数据与对象紧密耦合,适应范围也有限,也可能由于复杂性较高而影响系统的整体性能。

2.4 企业业务的划分问题

面向对象技术在EIS中应用的另一个常见问题,是如何析取对象及对象的方法。通常是按功能职责将企业的相关业务归集为某一对象的方法,即企业业务的划分是通过方法向对象的归集来实现的。但是,方法的归集对象不同,不仅会影响到现实业务职责表示的明确性,也会反过来影响到对象的设计、数据表示和应用等。例如,对于数字化医院软件中的门诊挂号业务,将"挂号"方法归集于患者对象既不确切,也给患者对象的持久化等问题的解决带来麻烦。当前的EIS研究中普遍缺乏对企业业务划分方法的研究,采用动态菜单或基于角色的业务划分等的处理机制[14,21]等则是对传统方法在应用上的扩展,并不能从根本上解决问题。

3 实体驱动业务信息系统框架EDTISF

3.1 EIS系统抽象及形式化定义

从组织结构和管理的视角来看,企业工作人员可看作是一些相对稳定的管理单元,而每个管理单元可以视为一系列管理与决策活动,这些活动的实质是在特定的管理思想和方法的指导与控制下对相关人员、物资、资金、信息等资源进行合理使用和调度[15]。例如,在数字化医院软件中,挂号员的职责是管理挂号单,挂号业务是由挂号员依据患者提供的信息对挂号单完成数据补充的操作。

由此,我们将企业中所有物理存在的和逻辑定义的管理对象(包括资源对象)都看作是操作员(或企业员工)管理的、相对独立的"受控实体"。管理对象在企业中通常具有按部门归集的特性,因此"受控实体"的分组及它们之间的层次关系可直接与企业管理的层次性组织结构相对应。企业员工(系统用户)是在获得受控实体的管理权限后,通过对相关受控实体的业务调度来完成工作任务。其中,业务是负责完成信息的采集、存储、操作和信息传递等的一个整体性数据处理过程,其结果能够导致受控实体的部分特性变化和业务部门间的迁移。为实现这一目标,业务对象需要描述其处理的数据对象集和数据处理步骤序列。一个数据处理步骤是一个单一

的数据处理过程，其上附有相应的输入/输出数据格式描述。数据对象对应用于企业信息资源，一是定义企业信息资源的组织结构，包括信息的数据项（字段名称、显示名称、类型、长度等）、数据源（数据库、数据表或视图名称、字段列表、代理用户信息等）和数据项的组合或拆分方式等；二是为其他对象提供一致性的数据访问途径。从而，EIS 可抽象化表示为“用户在安全控制下对受控实体的一组业务集合的调度”，其形式化定义如下：

• EIS 由用户集 U、受控实体集 E、任务集 T、数据对象集 D、单一数据处理过程集 P 和权限集 R 等基本构成元素；

• 相关对象间的关系均是多对多的关系，即有：$UA\subseteq U\times E$、$TA\subseteq T\times E$、$PA\subseteq P\times T$ 和 $DA\subset D\times E$；

• 用户可以获取数据对象的部分权限，即 $RA\subseteq U\times D\times R$；

• $\forall p_i,\exists t_i e_i(t_j\in T\wedge e_j\in E\wedge(t_j,e_j)\in TA\wedge(p_i,t_j)\in PA)$；

• $uter: p\rightarrow U$，将一个处理步骤映射到一个单一的用户，且 $uter(p_i)\in\{u(u_i,e_j)\in UA\}$；

• $data\_objgct: pc\rightarrow 2^D$，将一个处理步骤映射到一组数据对象 $data\ object(p_i)\subseteq\{d|(d,e_j)\in DA\}$，且其操作权限为 $\bigcup_{dsdra\_\_ltat(p_i)}\{r|(usgr(p_i),d,r)\in RA\}$。

3.2 EDTIS 框架体系结构

依据上述 EIS 的抽象化表示和定义，我们提出了实体驱动业务的信息系统框架。如图 1 所示，EDTISF 从上至下依次划分为应用层、表示层和数据层等 3 个基本层次，各层及其子层可依据实际应用设计进一步细化。

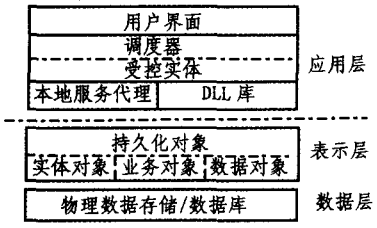

图 1 EDTISF 体系结构

1）应用层

应用层由用户界面、以受控实体为中心的调度器、本地服务代理和 DLL 库等功能子层构成。应用层通过受控实体的业务调度平台集中表示 EIS 的系统功能。应用层的核心是调度器，负责接收操作员对受控实体的业务调度指令，创建受控实体对象并通知其启动指定的业务。本地服务代理是为简化应用而对表示层服务进行的一致性接口封装，负责向表示层迁移受控实体和业务对象的数据存取任务，必要时完成数据的本地化转换等任务。操作代码库 DLL 存储具体的业务处理步骤过程，在收到业务的操作请求后，完成操作代码装载和自动运行等任务。用户界面对应于受控实体业务操作的界面需求，为操作员提供数据和业务流程控制的交互平台。

2）表示层

表示层的主要任务是向应用层提供持久化对象的表示服务，包含一组独立或关联的数据对象或数据对象协调器。这里的数据对象具有近似于服务组件的特性，但并不负责数据的本地化转换任务。数据对象收到其他对象的请求信息后，查找数据源，建立连接和存取数据，并依据系统对企业信息资源的组织定义，完成格式转换或数据重组等任务，即表示层数据对象的输出数据具有全局一致性的格式和组织形式。表示层的另一个功能是负责受控实体及业务对象的持久化表示。受控实体和业务对象并不包含具体的处理数据，仅仅是描述其需求的数据。

3）数据层

数据层是可共享的企业信息存储体系（包括本地的、既有应用系统中的和共享机构中的等），由于其涉及的技术相对比较成熟，这里我们不再叙述。

3.3 框架实现的关键技术

受控实体及其业务定义在 EIS 中具有相对稳定的特性，易于发生变化的是业务操作过程及其数据构成和表现形式。因此，EDTISF 实现的重点在于普遍适应的业务调度机制和灵活的数据对象表示方法，即应用层和表示层的设计与实现。如图 2 所示，EDTISF 实现中的操作员对象（类似于角色定义）是对系统用户按企业员工业务职责的一种划分。系统配置是为开发/软件维护人员提供一个 EDTISF 系统对象管理工具，负责各对象的定义及对象间的关系设定等任务。

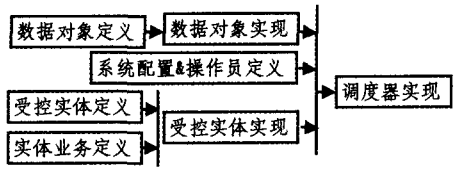

图 2 EDTISA 开发模型

3.3.1 受控实体及其业务组定义

框架开发过程中，受控实体和业务定义的主要是完成实体和业务定义，以及它们的组织方式和存储结构定义等任务。依据受控实体定义和企业资源对象往往具有比较明显的部门归属特性，如处方和系统用户等。定义受控实体的组织结构如下：

EC={ID,Name,Type,FGID,Description}；

Type=[Entity|Group]。

其中 FGID 与企业的部门代码对应，其取值为 0 时，表示该受控实体为一公用实体或实体组，否则表示为确定的部门或实体组中的子组/实体。

实体业务定义则主要是完成业务及其操作序列的组织结构描述和调度接口定义等。

3.3.2 数据对象

```
<DataObject ID=DO1011 name=chufang>
  <Title>住院西药处方</Title>
  <Section name=hzXinxi>
    <Item source= dbZhuyuan TableName=hz_zyxx>
      <Field=zyh>住院号</Field>
      <Field=ryrq format="yyyy-nn-dd hh:mm:ss">入院日期</Field>
      …… ……
    <Item>
    <Item source=dbZhuyuan TableName=hz_jbxx>
      <Field=xingming>患者姓名</InfoMeta >
      …… ……
    <Item>
  </Section>
  <Section name=cfNeirong>……</Section>
  <Section name=ysXinxi>……</Section>
  <Section name=sfXinxi>……</Section>
</Segment>
```

图 3 处方数据对象定义

数据对象定义包括两部分：1）数据来源描述，相关 EAI

的研究成果中有较多的研究成果，如文[3,13]等，这里不再叙述；2)结果数据的应用特性说明，如字段的显示名称和分解/组合应用规则等，其目的是为了能够使各个业务对象自主解析和正确使用结果数据。图3给出了处方数据对象定义的部分信息。由于数据对象是负责为业务对象提供透明的企业信息资源的数据表示服务，因此EIB技术非常有益于此部分的功能实现。

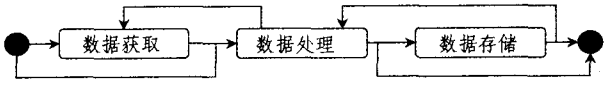

图4 单个实体业务的执行过程

3.3.3 受控实体实现

从EDTISF的体系结构可知，受控实体对象实现的关键是其业务步骤的调度机制。如图4所示，受控实体业务一个典型的事务流程包括数据获取、数据处理和数据存储3个步骤，该流程的非线性执行表现在数据获取与数据存储可能需要用户介入控制来完成。EDTISF将数据获取和数据存储交由数据对象完成，数据处理的过程代码则脱离业务对象，独立存储于DLL库中。DLL文件的导出函数定义为：

function Create(Task: TTask; OpID: Integer; UserID: String): IStep;

其中，UserID是执行业务的操作员，Task是步骤代码可用的业务对象，OpID则是要执行的操作序号，函数返回值为IStep接口对象，其方法定义如下：

```
IStep=interface(IInterface)
    function RunStep: boolean;
    function Abort: boolean;
    ……
end;
```

业务对象在相应的事件处理过程中调用Istep接口的方法完成其业务调度。业务对象可能包含多个数据处理步骤，这些处理步骤的执行序列通常是由顺序、分支和循环等几种基本执行结构组合而成的[3,26]。因此，业务对象的数据处理可简单描述为对其既定执行序列的解释和处理步骤的执行请求，这里简化为对一个DLL函数的调用。

4 比 较

CBSA、SOA等技术的核心是行业业务功能单元的可重用性构造。数据表示等中间件技术是普遍采用的一种企业异构数据源数据集成的有效方案。EDTISF侧重于研究如何有机划分企业的业务集，“受控实体”概念的提出，使得企业的业务划分更加确切和有利于EIS职责分离，以及对EIS的业务调度和数据应用方式进行一致性表示。有别于CBSA和SOA中的构件和服务单元，EDTISF的业务并不实际包含数据，而仅仅是描述需求的数据和解析数据对象的返回结果，从而进一步降低了业务和数据间的耦合度。从开发过程而言，CBSA和SOA等是在不同层次上对既有的构件和服务单元的客户化定制，EDTISF则只是客户化的系统配置定义，重点是系统分析期的受控实体及其业务归属和数据对象的定义与实现。CBSA和SOA等应用的结果是新的构件和服务单元，EDTISF不产生新内容，但可以作为一种生产业务、数据构件或服务单元的有效手段。相对于其它EIS开发技术，EDTISF的优势在于：

1)基于EDTISF完成的EIS软件能够支持业务的即插即用，依据企业的业务扩展和调整，通过系统配置工具，可方便插入/卸载受控实体及其业务和数据对象的定义，再加上其对版本控制技术的支持，能名使EIS具有更好的伸缩性和可扩展性。

2)EDTISF是在应用级别上对业务划分和调度机制的重用，与具体的应用环境无关，因此有更普遍的适用性。业务对象与数据对象的分离，降低了业务与数据的耦合度，增强了EIS对企业信息动态性的应对能力。首先，业务与数据对象的重组/变更简单易行；其次，数据描述/表示方法不变时，业务对象和数据对象的变化不会相互影响，数据描述/表示方法变化时，也只是需要修改全局统一的解析算法，从而使应用系统在不断进化中趋于完善。

3)各业务系统统一表现为按业务部门对“受控实体”进行分层组织管理和业务调度，有利于简化传统MDI和SDI等用户界面，能更好地支持动态菜单/窗体加载等界面控制技术。

4)支持多种软件开发模型和方法。

此外，相对构件连接器的设计和实现更加简单的业务调度机制，降低了EDTISF框架开发的难度，而框架一次开发重复使用的特性，也能使EIS软件开发人员的精力更加集中于企业业务的分析与设计，加速EIS软件开发进程。

结束语 针对目前的SA和ISA中普遍缺乏企业的业务划分规则，依据对EIS的抽象化表示：“操作员在安全控制下对受控实体的一组业务集合的调度”，和融合多种EIS开发技术、思想，提出了一种实体驱动业务的信息系统软件框架EDTISF。EDTISF通过引入“受控实体”的概念，实现了企业业务的确切划分，采用融合了企业信息总线概念的数据对象实现企业信息的集中标准化封装和应用系统的自主解析，有效降低了业务及其处理数据的耦合度，并在应用级的层次上实现了功能模块调度机制的重用，为EIS软件开发提供了一种有效的分析、设计和实现方案，目前该结构已应用于数字化医院软件和高校学生信息管理系统的设计与实现。

后续研究工作除了进一步完善EDTISF的方法论外，还将在EDTISF数据对象表示方法的简化，业务调度机制的优化，层次结构的形式化定义和访问控制技术等方面继续研究工作。

参 考 文 献

1 林东豪，高济，严国平. 支持敏捷制造的企业新型信息系统体系结构研究. 浙江大学学报(工学版)，2001，35(1)：77～82

2 Xu W，Yin B L，Li Z Y. Research on the business component design of enterprise information system. Journal of Software，2003，14(7)：1213～1220

3 刘卫东，宋佳兴，方晓彤. 一种信息系统集成方法. 计算机科学，2005，32(5)：67～70

4 Johnson R E. Frameworks Equal(Components + Patterns). Association for Computing Machinery. Communications of the ACM，1997，40(10)：39～42

5 Karhunen H，Jantti M，Eerola A. Service-oriented software engineering (SOSE) framework. In：Services Systems and Services Management，2005. Proceeings of ICSSSM '05. 2005 International Conference on Volume 2，June 2005. 1199～1204

6 Gold N，Mohan A，Knight C，et al. Understanding service oriented software. Software(IEEE)，2004，21(2)：71～77

7 Zachman A J. A Framework for Information Systems Architecture. IBM Systems Journal，1999，31(3)：445～470

(下转第167页)

(3)在利用 agents 进行数据挖掘时采用何种协作方式来获得较好的全局结果。

(4)局部节点的知识是否都能用于全局结果的构成,如何消除其中全部不感兴趣因素(这也就是如何给各个节点以挖掘目标的问题)。

致谢 本文的形成得益于周志华教授的《数据挖掘》课程的学习以及他的指导。

参 考 文 献

1 Han J, Kamber M. Data Mining: Concepts and Techniques. Morgan Kaufmann, 2000
2 Fu YongJian. Distributed Data Mining: An Overview [B]. IEEE TCDP newsletter, Spring, 2001
3 Park B H, Kargupta H. Distributed Data Mining: Algorithms, Systems and Applications. Data Mining Handbook, 2002
4 Provost F. Distributed Data Mining: Scaling up and Beyond. In: Kargupta H, Chan P, eds. Advances in Distributed and Parallel Knowledge Discovery AAAI Press / The MIT Press, 2000
5 Nwana H S. Software Agents: An Overview. Knowledge Engineering Review, 1996, 11
6 Jennings N R, Wooldinge M J. Agent Technology-Foundation, Application and Markets, Springer, 1997
7 Martin D L, Cheyer A J, Moran D B. The Open Agent Architecture. A Framework for Building Distributed Software Systems [J]. Applied Artificial Intelligence: An International Journal, 1999, 13 (1-3): 91～128
8 de Meo P, Rosaci D, Sarne G M L, et al. An XML-based Adaptive Multi-agent System for Handling E-commerce Activities. ICWS-Europe 2003, LNCS 2853, Berlin. Springer-Verlag, Heidelberg, 2003. 152～166
9 Huang Z, Eliens A, Visser C. 3D Agent-based Virtual Communities. In: Wagner W, Beitler M, eds. Proc. Int Web3D Symposium, ACM Press, 2002. 137～144
10 CHEN Gang. A distributed data mining system based on agent. Computer Engineering, 2001, 27(9): 65～68(in Chinese)
11 Klusch M, Lodi S, Moro G. Agent-based Distributed Data Mining: The KDEC Scheme. In: AgentLink, volume 2586 of LNCS. Springer, 2003
12 Zhang Z, Zhang C, Zhang S. An Agent-Based Hybrid Framework for Database Mining. Applied Artificial Intelligence, 2003, 17(5-6): 383～398
13 Davies W H E, Edwards P. Agent-based Knowledge Discovery [C]. In: AAAI Spring Symposium on Information Gathering, 1995
14 Davies W, Edwards P. Distributed learning: an agent-based approach to data-mining. R. Scotland, AB9 2UE: University of Aberdeen, 1997
15 Bailey S M, Grossman R L, Sivakumar H, et al. Papyrus: A System for Data Mining over Local and Wide Area Clusters and Super-Clusters: [Technical report]. University of Illinois at Chicago
16 Kargupta H, Park B, Hershberger D, et al. Collective Data Mining: A New Perspective Toward Distributed Data Mining. In: Advances in Distributed and Parallel Knowledge Discovery, AAAI/MIT Press, 2000. 131～178
17 Kargupta H, Hamzaoglu I, Stafford B. Scalable distributed data mining using an agent based architecture [A]. In: Proc. of KDD97 [C]. Menlo Park, CA: AAAI Press, 1997. 211～214
18 Stolfo S J, Prodromidis A L, Tselepis S, et al. JAM: Java agents for meta-learning over distributed databases. In: Hecherman D, Mannila H, eds. Proc. Third International Conference on Knowledge Discovery and Data Mining(KDD-97), Newport Beach, California, USA, AAAI Press, 1997. 74～81
19 Prodromidis A L, Chan P K, Stolfo S J. Meta-learning in Distributed Data Mining Systems: Issues and Approaches. In: Advancesin Distributed and Parellel Knowledge Discovery. Chapter 3. 2000
20 Rana O F, Walker D W, Li Mao-zhen, et al. PaDDMAS: Parallel and distributed data mining application suite [A]. In: Proc. 14th Int Conf. on Parallel and Distributed Processing Symposium [C]. Cancun Mexico: IEEE, 2000. 387～392

(上接第 135 页)

8 Wagner G. Toward Agent-Oriented Information Systems: [Technical report]. Institute for Information, University of Leipzig, March 1999
9 McDonald J T, Talbert M L, Deloach S A. Heterogeneous Data-Base Integration Using Agent-oriented Information Systems
10 马振宇,黄启春. 支持分布应用的可组态管理信息系统体系结构. 大连理工大学学报,2003,43(S1):118～122
11 石双元,陈琼,吴新明. 基于构件的信息系统开发框架. 计算机工程与科学,2004,26(10):106～109
12 张明宝,夏安邦. 基于面向服务体系架构的敏捷虚拟企业信息系统框架. 计算机集成制造系统,2004,10(8):985～990
13 王颖,吴荣泉,黄美锋,等. 一个面向服务的 EAI 框架. 计算机工程,2006,32(1):279～281
14 Yu Ming-Hui, Fei Qi, Chen Xue-Guang. A Role Interactive Model Based on Method of information System Function Partition. Science & Technology Progress and Policy, 2004, 9: 105～107
15 Ren Ming-Lun, Zhu Wei-Dong, Yang Shan-Lin. Component Based Architecture of Information Systems. Mini-Micro Systems, 2004, 25(7): 1159～1163
16 Feng C, Qianxiang W, et al. An architecture-based approach for component-oriented development. In: Computer Software and Applications Conference, 2002. COMPSAC 2002. Proceedings. 26th Annual International, Aug. 2002. 450～455
17 Succi G, Pedrycz W, etc. Package-oriented software engineering: a generic architecture. IT Professional, 2001, 3(2): 29～36
18 Kirk D, Roper M, Wood M. Defining the problems of framework reuse. In: Computer Software and Applications Conference, 2002. COMPSAC2002. Proceedings. 26th Annual International, Aug. 2002. 623～626
19 Chaudet C, Greenwood R M, et al. Architecture-driven software engineering: specifying, generating, and evolving component-based software systems. Software(IEEE), 2000, 147(6): 203～214
20 Fontaine A, Truong Anh-Thu, Manley T. A Survey of Strategies for Object Persistence. CSci 5802-Spring, 2006
21 Chai Yue-ting, Zhang Xiao-dong, Dong Jin. Research on the reconstructivity of agile supply chain management system. Journal of Tsinghua University, 2000, 40(3): 68～71
22 Nikolaidou M, Alexopoulou N, Tsadimas A, et al. A Consistent Framework for Enterprise Information System Engineering, 2006. http://www.hua.gr/Pythagoras Website/Docs & PDF/A Consistent Framework for Enterprise Information System Engineering.pdf
23 Zachman A J. A Framework for Information Systems Architecture. IBM Systems Journal, 1999, 31(3): 445～470
24 Wang Zhongjie, Xu Xiaofei, Zhan Dechen. A Survey of Business Component Identification Methods and Related Techniques. International Journal of Information Technology, 2005, 2(4): 229～238
25 PENG Yan-hong. Study on Enterprise Information Systems Architecture. Journal of System s Science and System s Engineering, 2000, 19(1): 77～86
26 张彬,李国辉,郭军. 一种基于分层平面的工作流组织模型的研究与实现. 小型微型计算机系统,2004,25(7):1253～1255