

企业信息系统业务构件设计研究*

徐 玮⁺, 尹宝林, 李昭原

(北京航空航天大学 计算机科学与工程系, 北京 100083)

Research on the Business Component Design of Enterprise Information System

XU Wei⁺, YIN Bao-Lin, LI Zhao-Yuan

(Department of Computer Science and Engineering, Beijing University of Aeronautics and Astronautics, Beijing 100083, China)

+ Corresponding author: Phn: 86-10-82317648, Fax: 86-10-82316100, E-mail: samuel_xuxu@hotmail.com

<http://www.buaa.edu.cn>

Received 2002-09-20; Accepted 2002-11-04

Xu W, Yin BL, Li ZY. Research on the business component design of enterprise information system. *Journal of Software*, 2003,14(7):1213~1220.

<http://www.jos.org.cn/1000-9825/14/1213.htm>

Abstract: Component-Based development method is thought to be the latest outcome to tackle software crisis, but in practice it didn't reach the expectation. The methodology of component design is not perfect due to the lack of explicit guidance to build the practical application. In this paper, a business component design method is given, which uses the Cluster Algorithm to identify two kinds of business components (process component and entity component) and takes the requirement models as the data source of Cluster Algorithm samples. Several formulations are also given to calculate the value of samples' relationship. "Core Entity" was chosen to achieve better accuracy of Business Component identification, and granularity formulation is given to reduce the complexity of Business Component. The design steps of Business Component Interface are described at last.

Key words: business component; information system; cluster algorithm

摘 要: 基于构件的软件开发被认为是解决软件危机的最新成果,但其实践并未达到预期效果:构件设计的方法学还不完善,缺乏对实际应用的明确指导.给出一种业务构件设计方法,应用聚类算法来识别两类业务构件(过程构件和实体构件),以需求模型作为聚类分析样本点的数据来源,并给出样本点关联值的计算方法.通过引入“聚集点实体”的概念改进识别业务构件的准确性,给出构件粒度的计算公式,以降低业务构件的复杂度.最后描述了业务构件接口的设计步骤.

关键词: 业务构件;信息系统;聚类算法

中图法分类号: TP311 文献标识码: A

* Supported by the National Natural Science Foundation of China under Grant No.79770007 (国家自然科学基金); the National High-Tech Research and Development Plan of China under Grant Nos.863-511-946-005, 863-511-920-002 (国家高技术研究发展计划(863)); the Key Science-Technology Project of the National 'Ninth Five-Year-Plan' of China under Grant No.96-B03-03-05-02 (国家“九五”重点科技攻关项目)

第一作者简介: 徐玮(1976—),男,福建莆田人,博士,主要研究领域为软件构件,数据库,信息系统,软件工程.

基于构件的软件开发(component-based software development,简称 CBSD)被认为是解决企业信息系统开发和维护问题的最新成果,近几年业界构件标准(如 EJB,COM+和 CORBA 等)的成熟进一步促进了 CBSD 的发展和应用^[1].但是,CBSD 尚未达到人们预期的目标:构件设计的方法学还不完善,缺乏对实际应用的明确指导.以 Business Component^[2],Catalysis^[3],UML Component^[4]等为代表的一些方法从完整的构件开发过程的角度,提到了一些基本的构件设计原则和方法.在此基础上,COMO^[5],O2BC^[6],Hemant^[7],Jong^[8]等人提出了 CRWD 矩阵法^[5-6]和聚类分析法^[7,8],用以识别构件,但并未充分考虑应用领域的特点.本文将重点讨论企业信息系统领域的业务构件设计问题.

本文首先分析基于构件的企业信息系统体系结构,并由此确定了两种业务构件(过程构件和实体构件).然后应用聚类分析的算法来识别业务构件,以需求模型作为样本点的数据来源,并给出样本点关联值的计算方法.为了提高识别业务构件的准确性,引入“聚集点实体”的概念对聚类分析算法加以改进,给出业务构件粒度的计算公式降低业务构件的复杂度.最后,描述了两种业务构件接口的设计步骤.

1 基于构件的企业信息系统体系结构

体系结构是软件开发过程的重要产品和系统性重用的主要资产,基于构件的软件开发必须以体系结构为中心.研究企业信息系统的构件体系结构有利于做出正确、稳定、灵活的系统设计,从而为构件设计的决策提供具体指导和框架支持.

层次体系结构是目前较为通用的企业信息系统构件体系结构之一^[9],它将系统的不同特征组织在不同层次上,促进了不同层次的软件重用.由此可以得出如图 1 所示的基于构件的五层体系结构,从上到下依次为应用接口层、业务过程层、业务实体层、构件容器层和平台软件层^[2,6].应用接口层由界面构件组成,它们对应于用户界面需求,并负责触发业务事件以启动业务过程.业务过程层由过程构件(process component,简称 PC)组成,对应于业务领域内重要的业务过程及相关业务规则.业务实体层由实体构件(entity component,简称 EC)组成,对应于业务领域内重要的业务实体及相关业务规则.构件容器层和平台软件层为上面 3 个构件层提供构件运行环境和基础设施.本文将重点论述利用需求模型识别和设计过程构件和实体构件的方法.

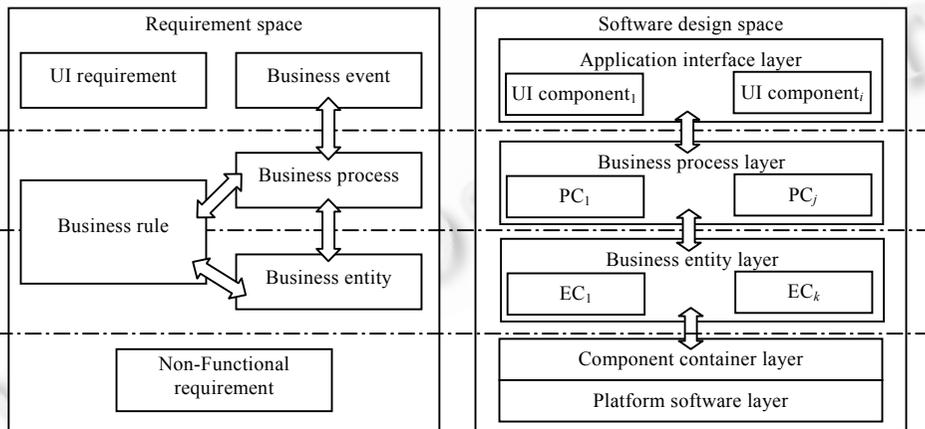


Fig.1 Component-Based architecture of enterprise information system

图 1 基于构件的企业信息系统体系结构

2 业务构件设计方法

业务构件设计既要满足系统需求,又要具备高内聚、低耦合、易重用的特点.目前通常采用 UML 语言来描述需求模型,它包括业务用例模型和业务实体模型两部分,其中的核心元素是业务用例和业务实体.因此,从业务用例和业务实体中识别出高内聚、低耦合的业务构件,是业务构件设计阶段的首要环节.构件是独立发布的封装体,其内部实现对外不可见,其接口是构件与其他构件和周围环境进行交互的惟一方式^[10],因此,接口(操作)

设计也是业务构件设计阶段的一个重要任务。

基于上述分析,我们给出“以识别构件和设计接口为中心的业务构件设计过程”,如图2所示。该方法以UML语言描述的业务用例模型和业务实体模型作为输入,通过改进的聚类分析算法识别出两类业务构件,进而设计出业务构件的接口,得出两类业务构件的设计规格说明,从而完成业务构件设计过程。

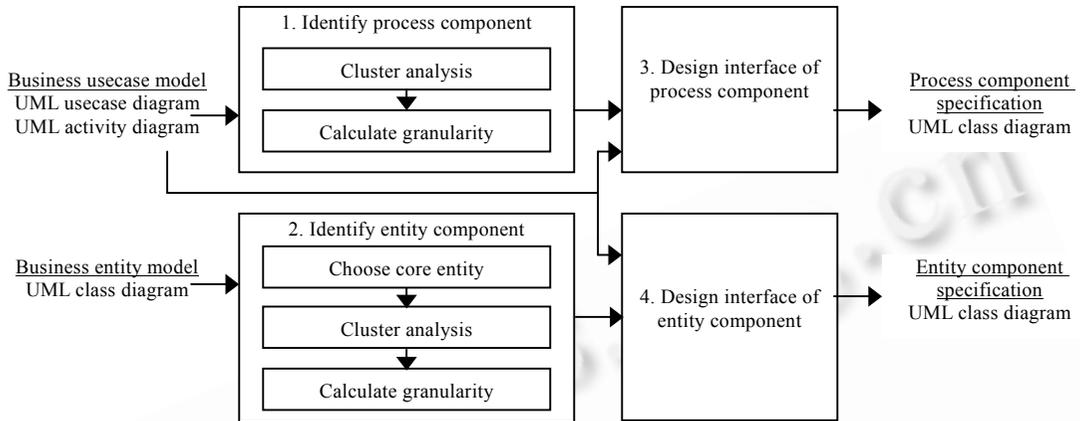


Fig.2 Business component design process

图2 业务构件设计过程

2.1 识别业务构件

2.1.1 聚类分析

聚类分析(cluster algorithm)^[11]是数理统计中研究“物以类聚”的一种方法。本文用以识别业务构件,即按照业务构件语义从需求模型中选择样本点集合进行分类。聚类分析的基本步骤如下:

- (1) 将所有需要分类的 n 个样本点记为 $X=\{X_l\}(l=1,\dots,n)$ 。此时,集合 X 中的每个样本点自成一类。
- (2) 确定任意两个样本点之间的关联值(即相似程度)的计算方法,将样本点 X_i 与 X_j 之间的关联值记为 R_{ij} 。
- (3) 计算 n 个样本点两两之间的关联值,得样本点间的关联值矩阵 D 。
- (4) 选定一个合理的最小关联值 R_{\min} ,将其作为将两个样本点放入一个聚集的判定依据。
- (5) 依据关联值矩阵 D 中每个单元的关联值 R_{ij} ,执行以下聚类过程:

- (阈值)如果 $R_{ij} \geq R_{\min}$,则将样本点 X_i 和样本点 X_j 放入一个聚集。
- (传递性)如果 X_i 和 X_j 属于一个聚集,且 X_i 和 X_k 属于一个聚集,那么将 X_i, X_j, X_k 放入一个聚集。

最后,可将每个聚集中的所有样本点合在一起转换为一个业务构件。

2.1.2 识别过程构件

聚类分析依赖于样本点集合的选取以及对样本点之间的关联值的理解,定义不同的样本点集合和关联度量度将产生不同的聚类结果。Hemant^[7]和 Jong^[8]以业务实体作为 X 样本点集合,以业务实体间关联的强度(relationship strength)作为关联值,并给出了若干计算公式。COMO^[5]和 O2BC^[6]采用了聚类分析的一种特例——CRWD 矩阵法,它们以业务用例和业务实体作为 X 样本点集合,以用例和实体的4种关系:创建、读取、写入和删除来计算关联值,具有“创建”或“删除”关系的用例与实体被合并到一个业务构件中。

下面首先阐述过程构件的聚类分析过程中的样本点选择和关联值计算的方法。从图1中可以发现,过程构件是对业务过程和业务规则的封装,采用UML用例图和活动图描述的业务用例是一种目前较为通用的描述业务过程的表示元素,因而选择以业务用例作为识别过程构件的 X 样本点集合。业务用例之间的关系主要有3种:包含(include)、扩展(extend)和泛化(generalization)^[12],按照其特点确定关联值 R_{ij} 的计算。本文设定为:泛化关系的 $R_{ij}=3$;扩展关系的 $R_{ij}=2$;包含关系的 $R_{ij}=1$;没有关系的 $R_{ij}=0$ 。 R_{ij} 值越大,表明两个业务用例的联系越紧密,适合于放在一个过程构件内部。

2.1.3 识别实体构件

实体构件是对业务实体及相关业务规则的封装,通常使用 UML 类图中的类来描述.因此,识别实体构件所需的 X 样本点集合应该以 UML 类图中的业务实体(类)为基础.Hemant 等人采用的聚类分析过程将各个业务实体看作等价元素,但是实际系统中的各个业务实体的重要程度显然不完全相同,如此做法将导致系统的重心分散,无法解决系统的关键问题.为此,我们对第 2.1.1 节所述的聚类分析步骤加以改进.

(1) 关联值计算公式的选择

业务实体并不是孤立存在的事物,它们通过相互合作以完成系统功能需求,因此,除了要通过 UML 类图计算实体间的静态关联值以外,还必须通过 UML 用例图来计算实体间存在的动态关联值.综合文献[7,8]的研究成果,我们给出业务实体关联值的计算公式如下:

• 静态关联值

$$RS_{ij} = \sum_{y \in Y} (U_{yi} \cdot U_{yj} \cdot W_y). \quad (1)$$

其中 Y 为实体间关系的集合. U_{yi} 为二值变量:如果关系 y 中包含实体(样本点) X_i ,则 U_{yi} 设为 1;如果关系 y 中不包含实体 X_i ,则 U_{yi} 设为 0. U_{yj} 与 U_{yi} 的取值方法相同. W_y 为该关系 y 所占的权重,根据关系 y 紧密程度设定:泛化关系, $W_y=5$;组合关联, $W_y=4$;聚集关联, $W_y=3$;有向关联, $W_y=2$;无向关联, $W_y=1$.

• 动态关联值

$$RD_{ij} = \sum_{z \in Z} (V_{zi} \cdot V_{zj} \cdot W_z). \quad (2)$$

其中 Z 为用例的集合. V_{zi} 为二值变量:如果用例 z 使用了实体 X_i ,则 V_{zi} 设为 1;如果用例 z 中不包含实体 X_i ,则 V_{zi} 设为 0. V_{zj} 与 V_{zi} 的取值方法相同. W_z 为用例 z 所占的权重,可根据其重要程度进行设定.

• 关联值

$$R_{ij} = RS_{ij} + RD_{ij}. \quad (3)$$

(2) 选择聚集点实体,修改关联值矩阵建立方法

在算法第 3 步中增加一步:从所有的业务实体中挑选出核心的、关键的业务实体,称之为“聚集点实体”,并修改“关联值矩阵 D ”的建立方法:以“聚集点实体”作为矩阵的一维,记为 $H=\{H_l\} (l=1, \dots, n)$;以“非聚集点实体”(除去聚集点实体之外的其他业务实体)作为矩阵的另一维,记为 $V=\{V_l\} (l=1, \dots, m)$;然后应用公式进行计算.

选择聚集点实体的过程实际上是一个划分业务实体主次关系的过程.在数据库设计的过程中经常会遇到主表和从表的概念,通常来说,主表和从表之间为“一对多”的关系类型,主表一端的关系多重性(multiplicity)为“1”,且往往更为重要.业务实体与数据库表都是业务过程中所涉及到的事物的一种表达方式,具有较大的相似性.因此,我们将 UML 类图中可惟一标识、且与其他实体的各种关系多重性(multiplicity)都不超过 1 的业务实体^[4]作为“聚集点实体”.此外,也可以采用按照用例的重要性来确定聚集点实体:如果某业务实体用于实现核心的功能用例,则将其作为“聚集点实体”的候选者.

(3) 修改聚类过程

由于关联值矩阵的建立方法不同,需要对算法第(5)步的聚类过程加以修改.由于聚集点实体非常重要且相对独立,所以我们将聚集点实体 H_i 作为候选的实体构件,接下来只需将非聚集点实体 V_j 分配给与之 R_{ij} 最大的那个聚集点实体 H_i .对于某些特殊情况:(a) 如果该实体 V_j 对应的所有 R_{ij} 值都小于 R_{\min} ,那么最好将该非聚集点实体单独作为一个实体构件;(b) 如果某个非聚集点实体 V_j 与两个聚集点实体 H_i 和 H_k 的关联值 R_{ij} 和 R_{kj} 相等,且都 $\geq R_{\min}$,则需分别计算 H_i 和 H_k 包含的非聚集点实体个数 n ,然后将 V_j 分配给 n 值更小的那个聚集点实体.

2.1.4 计算业务构件粒度

聚类分析是一个将样本点个数由多变少的过程,设置不同的 R_{\min} ,不断地执行聚类会带来业务构件粒度的随之增大.为了保证识别出的业务构件不至于太复杂,有必要通过定量公式计算业务构件的粒度,以保证得出合理的构件设计结果.过程构件的粒度计算离不开业务用例.业务用例的内部实现流程通常采用活动图描述,经验表明,过程构件粒度随着活动图中的分支数和活动数这两个因素而线性增加.在软件度量领域,McCabe 提出的圈数复杂度度量法(cyclomatic complexity metrics)依据流程判断语句(do while, If then else)将整个程序模块划

分为多个区域,并将区域的总数作为程序的复杂度^[13].由此,我们提出如下过程构件粒度的计算公式:

$$Granularity(PC) = \sum_{u \in PC} [\alpha \times Branch(u)^2 + \beta \times Activity(u)]. \quad (4)$$

其中, PC 为某过程构件, $Granularity$ 为该构件的粒度, u 为过程构件 PC 所聚集的某一用例, $Branch(u)$ 为该用例的活动图中的分支数, $Activity(u)$ 为该用例的活动图中的活动数, α 为分支数所占的权重比例, β 为活动数所占的权重比例.根据实际应用经验,我们认为,对于一般的业务用例,合理的 α 值为0.6, β 值为0.4;但是,对于某些分支条件很多(大于7)的业务用例,如合同审批等,可将 α 值提高为0.7, β 值降低为0.3,以更好地反映实际特征.

实体构件的粒度计算相对简单,我们直接以实体构件包含的业务实体个数作为其粒度.

各个开发组织可根据应用领域、系统规模、技术积累等实际情况,确定业务构件计算公式的权重比例以及合理取值范围.当某个过程构件的粒度超出了合理取值范围时,可以采用 John Daniel 等人提出的方法^[4]:使用活动图的分支点来拆分用例,然后重新执行聚类分析过程.当实体构件的粒度超出合理取值范围时,可以适当增大 R_{\min} 的数值,以使聚类分析过程可以产生更多的实体构件.通常来说,一个中等规模的企业信息系统的业务构件数目在30~100个以内较为合适^[9].

2.2 设计业务构件接口

业务构件接口由一组相关的操作组成,因此,设计业务构件接口的主要工作是设计出一组可准确表达构件功能的相关操作.

2.2.1 设计过程构件接口

过程构件对应着业务用例,而业务用例内部实现通常采用活动图来描述,因此过程构件的接口定义应该能够表达出整个活动图描述的业务流程.对活动图中包含的各种图元进行分析之后,可以发现将每个“活动”图元映射为一个操作较为简单、有效,最后将识别出的所有操作合并为过程构件的一个接口^[4].此外,如果存在若干个活动必须满足一个事务完整性要求,最好将它们合并为一个操作,这样可以降低构件的实现难度^[4].这种设计方法使构件接口上的操作与企业业务模型中的业务活动得以直接对应,提高了软件的可追溯性.

2.2.2 设计实体构件接口

实体构件封装了企业业务模型中的业务实体,所以必须从业务实体完成的功能入手,找出实体构件的接口和操作.在前述的体系结构图中,业务实体并不直接与外部角色进行交互,它的主要功能是为上一层的过程构件提供业务实体的信息.因此,实体构件的接口设计依赖于业务功能层中的过程构件对业务信息的需求.设计实体构件接口可按如下方法进行:(1) 针对过程构件接口的每个操作的功能,设计过程构件与实体构件之间的交互图,采用“消息”方式来表达构件交互.尽量采用已有的设计模式,以获得最优构件接口设计结果.其中,构件交互图^[4]是 UML 交互图的扩展.(2) 将构件之间的每个消息调用转换成接收该消息的实体构件的一个操作.(3) 将多个构件交互图中获得的所有操作汇总起来,消除命名冲突,如同名不同义或同义不同名等,获得最终的实体构件的操作集.(4) 为每个实体构件定义一个接口,将该实体构件的所有操作放入该接口中.

3 实例

以某企业的库存子系统为例,介绍本构件设计方法的应用过程.该子系统的主要功能是完成器材的收发存管理和质量控制,并为其他子系统提供各种器材的库存数据.

3.1 识别过程构件

库存子系统的业务用例模型如图3所示.我们在实际聚类分析时,只将具有泛化关系的用例合并在一个构件内,因此设定 R_{\min} 为3.对图3中的所有用例执行聚类分析:“器材价格核算(price check)”、“计划价核算(plan price check)”、“实际价核算(real price check)”这3个用例关系为“泛化”,它们的关联值 $R_{ij}=3, \geq R_{\min}$,可将这3个用例识别为1个“器材价格核算”过程构件;“不合格器材退库(eject material)”用例与其他用例间最大的关联值 $R_{ij}=2, < R_{\min}$,因此将该用例识别为1个“不合格器材退库”过程构件.同理,可识别出其他过程构件.

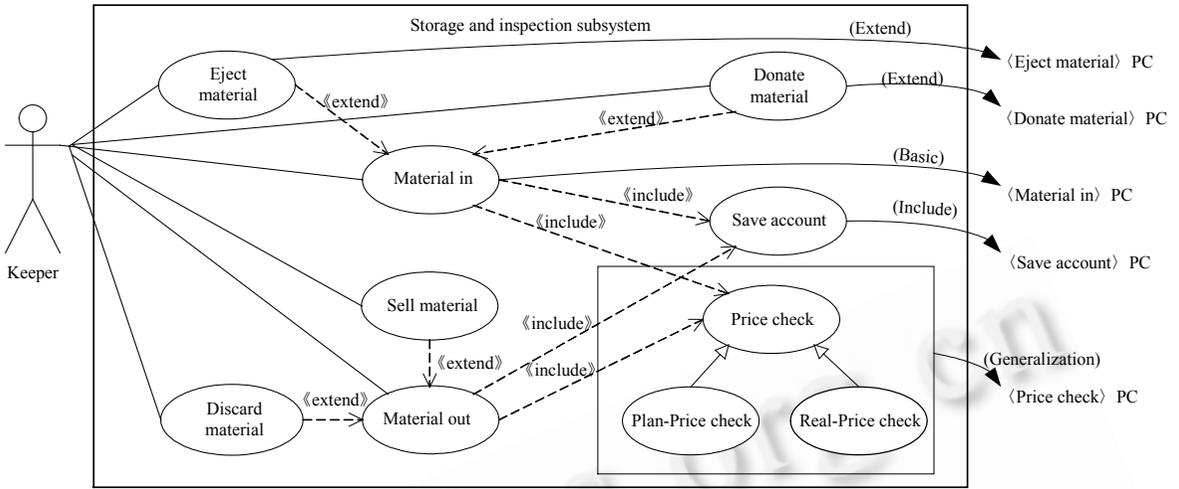


Fig.3 Identify process component

图 3 识别过程构件

3.2 识别实体构件

库存子系统业务实体模型如图 4 所示.按第 2.1.3 节所述,首先找出两个聚集点实体:C2 和 C5.虽然 C6 存在 1 条关联多重性为*的关系,但由于它在多个核心业务用例中都被使用,所以将其也作为 1 个聚集点实体.然后建立聚集点实体与非聚集点实体的关联值矩阵,见表 1.在实际聚类分析时,我们将 R_{min} 设定为 6.C1 与 C2 之间的关联值为 $5+3=8(\geq R_{min})$,C1 与 C5,C6 之间的关联值分别为 3 和 2,因此,将 C1 分配到 C2 聚集点实体.同理,可将其他非聚集点实体进行分配.最后,以聚集点实体为中心进行识别,可将 C2,C1,C3 识别为 1 个实体构件,C5,C8 识别为 1 个实体构件,C6,C4,C7 识别为 1 个实体构件.

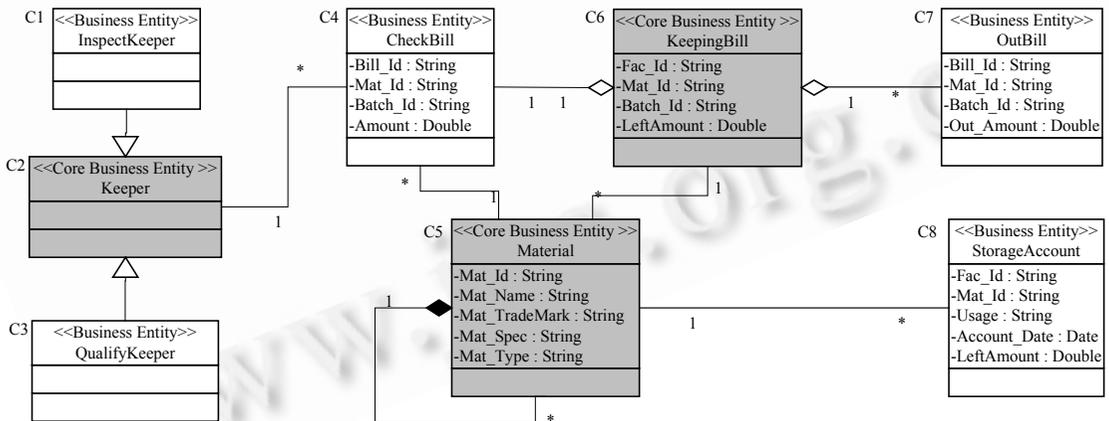


Fig.4 Identify entity component

图 4 识别实体构件

Table 1 Cluster algorithm matrix of business entity
表 1 业务实体的聚类分析矩阵

RS_{ij}, RD_{ij}		Non-Core entity				
		C1	C3	C4	C7	C8
Core entity	C2	5,3	5,5	3,2	0,3	0,2
	C5	0,3	0,4	1,2	0,3	1,5
	C6	0,2	0,4	4,2	4,3	0,4

3.3 设计构件接口

按第 2.2 节所述步骤,如图 5 和图 6 中箭头所示,可逐步地设计出业务构件的接口。

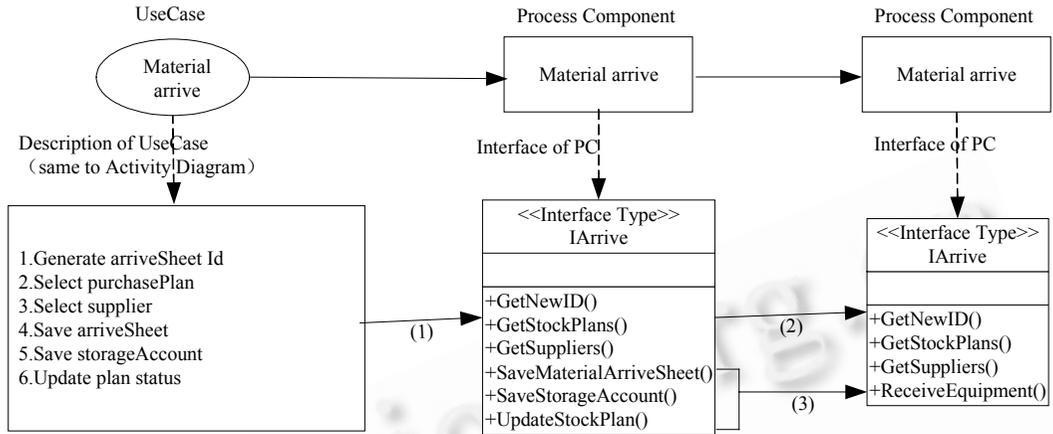


Fig.5 Design process component interface
图 5 设计过程构件接口

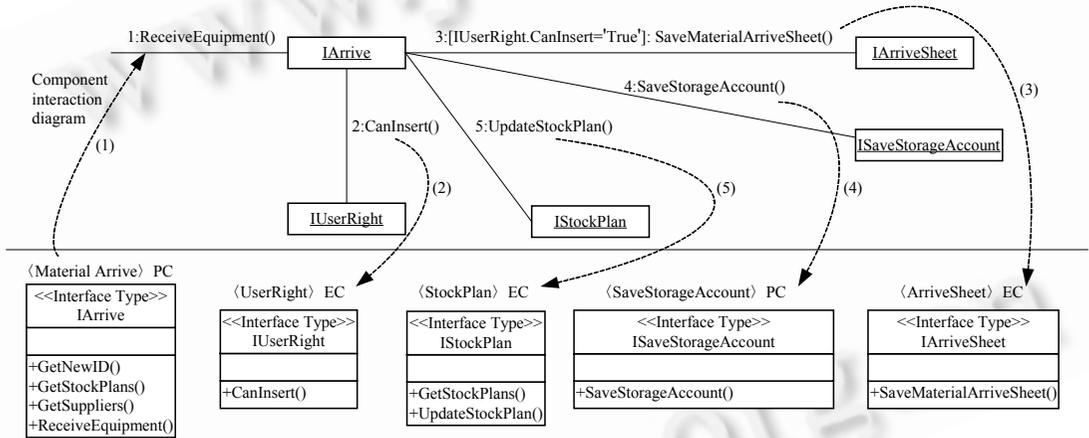


Fig.6 Design entity component interface
图 6 设计实体构件接口

采用本文所述的业务构件设计方法,共设计出 10 个业务构件,组装成为库存子系统的业务处理软件,它构成了该企业物资供应系统中应用服务器构件的重要部分。

4 结束语

本文采用聚类分析方法,通过对样本点的选择和样本点关联值的计算,提供了一种识别和设计业务构件的客观标准.经过在其他子系统中的应用,证明该方法设计出的构件更合理,复杂度恰当,具有较好的应用价值.目前,我们正在设计实现业务构件设计工具,以进一步提高业务构件识别的效率.此外,还将研究如何将 OCL(对象约束语言)应用于业务需求描述,从而为接口函数的详细设计提供基础.本文的构件设计方法仅完成了从概念层到设计层的映射,今后还将研究如何将构件设计结果映射到实现层。

致谢 在此,我们向对本文的工作给予支持和建议的同行,尤其是参与讨论的安琪同学和 Yahoo.ObjectiveView 讨论组的同行们表示感谢。

References:

- [1] Crnkovic I. Component-Based software engineering—New challenges in software development. *Software Focus*, 2001,2(4): 127~133.
- [2] Herzum P, Sims O. *Business Component Factory*. New York: John Wiley&Sons, Inc., 1999. 425~529.
- [3] D'Souza DF, Wills AC. *Objects, Components, and Frameworks with UML—The Catalysis Approach*. Reading: Addison-Wesley Longman, Inc., 1998. 505~680.
- [4] Cheesman J, Daniels J. *UML Components: A Simple Process for Specifying Component-Based Software*. Boston: Addison-Wesley Longman, Inc., 2000.1~65.
- [5] Lee SD, Yang YJ. COMO: A UML-based component development methodology. In: *Proceedings of the 6th Asia Pacific Software Engineering Conference*. Takamatsu: IEEE Computer Society Press, 1998. 54~63. <http://www.computer.org/proceedings/apsec/0509/0509toc.htm>.
- [6] Ganesan R, Sengupta S. O2BC: A technique for the design of component-based applications. In: *Proceedings of the 39th International Conference and Exhibition on Technology of Object-Oriented Language and Systems*. IEEE Computer Society Press, 2001. 46~55. <http://www.computer.org/proceedings/tools/1251/1251toc.htm>.
- [7] Jain H, Chalimeda N, Ivaturi N, Reddy B. Business component identification—A formal approach. In: *Proceedings of the 5th IEEE International Enterprise Distributed Object Computing Conference*. Seattle: IEEE Computer Society Press, 2001. 183~187. <http://www.computer.org/proceedings/edoc/1345/1345toc.htm>.
- [8] Lee JK, Jung SJ, Kim SD, Jang WH, Ham DH. Component identification method with coupling and cohesion. In: *Proceedings of the 8th Asia-Pacific Software Engineering Conference*. Macau: IEEE Computer Society Press, 2001. 79~88. <http://www.computer.org/proceedings/apsec/1408/1408toc.htm>.
- [9] Zhu SY, Qian LQ, Su WM. *Software Engineering Technology Conspectus*. Beijing: Science Press, 2002. 168~193 (in Chinese).
- [10] Szyperki C. *Component Software: Beyond Object Oriented Programming*. Harlow: Addison-Wesley Longman, Inc., 1997. 1~70.
- [11] Gao HX. *Practical Statistical Approach and SAS System*. Beijing: Peking University Press, 2001.185~254 (in Chinese).
- [12] Liu C, Zhang L. *Visualization Object-Oriented Modeling Language*. Beijing: Beijing University of Aeronautics and Astronautics Press, 1997. 33~99 (in Chinese).
- [13] Poulin JS. *Measuring Software Reuse-Principles, Practices, and Economic Models*. Reading: Addison-Wesley Longman, Inc., 1996. 109~134.

附中文参考文献:

- [9] 朱三元,钱乐秋,宿为民.软件工程技术概论.北京:科学出版社,2002.168~193.
- [11] 高惠璇.实用统计方法与 SAS 系统.北京:北京大学出版社,2001.185~254.
- [12] 刘超,张莉.可视化面向对象建模技术.北京:北京航空航天大学出版社,1997.33~99.