计算机科学1995 Vol. 22 №. 2

面向对象的MIS开发方法和平台*)

诸葛海

(第二炮兵第三研究所　北京100085)

摘　要　In this paper, an object-oriented analysis and design symbolic tool and related methodology is proposed, and a MIS development platform is also presented.

关键词　Object-oriented, Software, Development methodology, Software Development platform.

一、引言

面向对象(Object-Oriented,简称OO)是一种对应于真实世界概念化的抽象思维方法。OO方法学(OO Methodology 简称OOM)是以OO基本概念、方法和理论为基础的一套原则、策略、工具和环境，以此可有效地建立系统的符号模型。

目前关于OO的研究尚存在三方面的不足：(1) OO语义理论；(2)简洁实用的OO分析设计方法；(3)直接提供对象语义的软件开发平台。

文[6]讨论了面向对象的概念体系语义，本文在此基础之上提出一套简洁实用的OO分析设计符号工具和建立在这套符号工具之上的一种OO MIS开发方法，论述了支持该方法的MIS开发平台。

二、面向对象的分析与设计方法

2.1　符号工具的定义

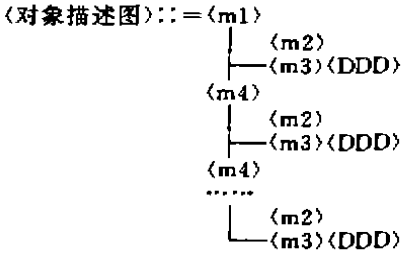
一套良好的符号(图形)工具是系统分析与设计的基础。结构化系统分析与设计(简称SA/SD)及其变形方法有一定局限性，如：①图形复杂，不便绘制、生成和交流；②文档庞大，导致长的开发周期；③分析和设计阶段分别采用两套不同的符号体系，使得分析的结果不便向设计阶段转化。④基于功能的方法难以适应领域发生的变化。为了克服这些不足，并与OO概念相衔接，下面提出有向目录图DDD，作为系统分析和设计的统一符号工具。

DDD的形式定义如下：

〈DDD〉::=△〈对象标识〉〈对象描述〉|△〈对象标识〉

〈对象描述〉::=〈对象描述图〉〈对象描述〉|〈空〉

〈对象描述图〉::=〈m1〉



其中，〈m1〉::=↑|〈m4〉　{↑表示继承关系}

〈m2〉::=a(〈原子对象〉)　{表示接收一个(触发状态转移的)原子对象}

|s(服务标识(〈描述〉))　{表示提供服务}

|c(服务标识(〈描述〉))　{表示引用服务}

|i(〈视图〉)　{表示继承箭头所指对象的一个视图}

|e　{表示一个实例对象}

|〈空〉

|m2〈标记〉

〈标记〉::=*　{表示多次重复}

|o　{表示可选}

|h　{表示所标记的对象被隐蔽}

|#　{表示关键对象}

|〈空〉

〈m3〉::=→　{状态转移关系}

|←→　{1:1关系}

|←→→　{1:m关系}

|←←→　{m:1关系}

|←←→→　{m:m关系}

|—　{一为图的连线}

〈m4〉::=↓!　{表示不能缺少的严格顺序关系}

*)本文得到国家八五科技攻关项目和国家自然科学基金的资助。诸葛海　博士后。

|↓	{表示非严格顺序关系}
|↓～	{表示层次结构}
||	{为图的无向连线}

加"△"的对象表示已定义的对象，未加"△"的对象表示可以继续用DDD逐层定义下去。"{}"中的内容表示注释，在文档的任意部分均可以插入用"{}"括起来的注释。DDD的定义允许任意嵌套。根据⟨mi⟩(i=1,2,3,4)的不同取值，DDD可表达出组成关系图、继承关系图、服务图、状态转移关系图、对象属性关系图、层次关系图及这些图的嵌套。本文所有的图均使用这种符号表示。

2.2 面向对象的分析(OOA)

系统分析的目的在于建立领域的对象模型，其结果应满足以下原则：1)可以模拟领域；2)便于交流；3)图形化；4)简洁性。

(1)OOA的内容和步骤 分析的工作原则是：1)自顶向下的分解；2)自底向上进行抽象。其步骤如下：

第一步：可行性分析。包括：划定系统范围(OOA对象的边界)，现行系统的缺点和新系统的目标，技术可行性和效益分析。

第二步：领域业务组成及服务分析。①依据第一步的系统范围和领域现行的管理体系自顶向下按业务及相关性进行分解，逐步将领域业务分解为独立的领域业务处理对象，直到不能再分的简单(业务)对象为止。在分解的同时构造出业务对象组成关系图，标识每个对象，描述其属性。将业务所涉及的实体也标识为对象，记所有业务和相关的实体集合为U_0。②自顶向下描述每个对象所提供的服务，构成服务组成图。每个服务都是以下的形式：

⟨服务⟩::=S(⟨服务标识⟩(⟨描述⟩)) {提供服务}
|C(⟨服务标识⟩(⟨描述⟩)) {引用服务}

⟨描述⟩::=(⟨接收对象部分⟩,⟨返回对象部分⟩,⟨服务说明⟩)

第三步：抽象。1)对象抽象：①将对象的属性抽象为属性型；②将对象的服务抽象为服务型；③将对象之间的关系作为对象抽象的关系对象型。2)型抽象：设O_i和O_j为两个对象，O_j提供服务$S_{j1},\cdots,S_{jn}$。O_i提供服务$S_{i1},\cdots,S_{im}$。S_{ik}和$S_{jk'}$分别为O_i和O_j的第k个和第k'个服务，$T(S_{ik})$,$T(S_{jk'})$分别为相应的型，$T(S_{ik})=T(S_{jk'})$，$1\leqslant k,k'\leqslant \min(n,m)$。其抽象策略如下：

①如果m=n，则形成O_{ij}，其服务与O_i及O_j的服务相同，否则进行②；

②抽取：形成新的O_{ij}，O_{ij}的属性和服务为两者的交集，并将O_i和O_j作为O_{ij}的子型。

第四步：合并优化。不受领域现行管理体系的限制，根据服务的相关性来聚合、组织对象。这一步将对象集U_0变换为新的对象集U_1，U_1就是以后步骤的基础。优化原则如下：

1)尽可能减少引用服务联结的复杂性，描述尽可能简单；

2)尽可能运用继承关系；

3)不含无用属性和服务。

第五步：对象继承描述。在U_1中描述每个对象自身特有的属性和服务，通过继承关系可得到的属性和服务不必重新描述，但要指出被继承对象的标识和视图，从而提高软件设计的可重用性。

(2)对象字典及辅助工具 对象字典(Object Dictionary，简称：OD)存放和管理对象及关系，定义如下：

⟨对象字典⟩::=(⟨对象表⟩,⟨服务表⟩,⟨属性表⟩,⟨参数表⟩)

⟨对象表⟩::=Object(⟨对象标识⟩,⟨对象名⟩,⟨对象型⟩,⟨服务链⟩,⟨组成链⟩,⟨属性链⟩,⟨状态转移链⟩)

⟨服务表⟩::=Ser(⟨服务标识⟩,⟨服务名⟩,⟨服务型⟩,⟨参数链⟩)

⟨属性表⟩::=Att(⟨属性名⟩,⟨属性型⟩,⟨值型⟩,⟨型长⟩,⟨初值⟩,⟨缺省值⟩)

⟨参数表⟩::=Para(⟨参数名⟩,⟨参数型⟩)

⟨值型⟩::=⟨整型⟩|⟨实型⟩|⟨字符型⟩|……

……

对于给定的领域，建立、维护和确认OD的过程实际上就是OOA的过程，OD应保持一致性和无冗余性。

OD辅助工具的主要功能是提供建立OD的界面，检查一致性，消除冗余，辅助抽象，生成五种形式的DDD。由于DDD是结构化的，消除了线条的交叉，所以给图的生成带来很多方便，同时也增强了图的可读性，简化了文档。

OOA的内容和步骤用DDD图描述于图1。

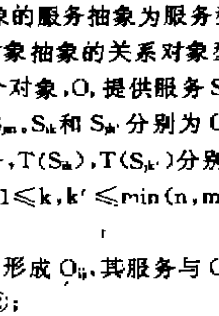


图1 OOA的内容和步骤

2.3 面向对象的设计(OOD)

OOD 的内容和步骤用 DDD 图描述于图2。

(1)确定 OOD 的对象　设:DS 为领域对象的服务类,SO 为系统对象类,SS 为系统服务类,确定 OOD 对象分 f1,f2两个映射来完成:f1:U_1→SO,f2:DS→SS。设计阶段的服务为如下形式:

〈服务〉::=S(〈服务标识〉(〈参数〉))|提供服务|
　　|C(〈服务标识〉(〈参数〉))|引用服务|
〈参数〉::=(In(〈参数〉),Out(〈参数〉),〈服务说明部分〉)

f2(〈接收对象〉)=In(〈参数〉)

f2(〈返回对象〉)=Out(〈参数〉)

f2(〈服务标识〉)=〈服务标识〉

在映射过程中要明确哪些对象及其服务是由软件系统来完成的,哪些不宜用软件系统完成要由人工或其它系统来完成。那些由软件系统来完成的对象就是 OOD 的对象。从这一步起,思维的角度要从"领域"转向"软件系统"。

(2)设计中的抽象　抽象是人类的智能行为,始终贯穿于 OOA/D 之中,抽象能力与人的知识和机制有关,是因人而异的,所以不同的设计人员设计出的软件系统存在差异。关于抽象的详细论述参见文[3]。

开发方法应包含策略和工具以帮助进行抽象。

OOD 的抽象目标:在 OOD 方法中定义了包括抽象数据类型在内的"预定义型",它给出对象抽象的目标以辅助设计者进行抽象,图3是一部分预定义型。

抽象策略:OOA 中的抽象方法和 OOD 的抽象方法是抽象方法的两个实例,换言之,OOA 的抽象方法完全可以在 OOD 中得到重新使用。

(3)软件服务组成　OOA 的组成及服务图经抽象后将其映射为预定义型和自定义型两部分。其策略如下:

①先描述 MIS 型,描述提供和引用的服务;

②考察 OOA 及服务图,选择 OOA 的组成及服务图中未考察过的对象;如果其领域服务可抽象为预定义型,则运用该预定义型,否则为自定义型,给出对象及服务的设计;标记对象为已考察过的;

③如果 f1(U_1)和 f2(DS)中的每个对象都已在软件服务组成图中描述过,则结束,否则继续进行②。

(4)软件对象继承图　软件对象的继承包括两层含义,一是通过 OOD 对象反映出的 OOA 所描述的领域对象和服务继承关系,二是 OOD 对象本身为了提高软件本身的可重用性而形成的继承关系。二者有一部分是一致的,但也有一部分不同。比如:有时为了提高效率而要增加冗余度。软件对象继承图按照对象型、子型和实例的继承关系分类描述。

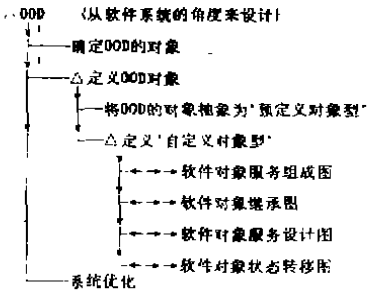


图2 OOD的内容和步骤

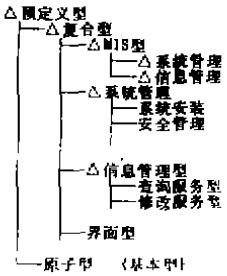


图3 预定义型

三、开发平台

与上述 OO 方法相衔接的应是一个可以直接提供对象语义的开放的 MIS 开发平台,在这个平台上开发者可运用所提供的丰富的开发手段方便、快速地开发出特定领域的 MIS。

3.1 总体介绍

(1)平台的结构　平台分为多个层次,为开发者提供多层次的服务,设计目标是标准化、开放性。平台的结构如图4所示,分五个层次向用户开放,由于目前的 UNIX 仍是 OS 的主流,具有丰富的软件,并将逐渐成为 OS 的标准,所以平台的第一层提供标准的 UNIX 程序员开发平台;第二层是 UNIX 上的标准 C 开发平台;第三层是标准的 SQL 开发平台;第四层是通用的4GL 开发平台;最高层是与 OO 概念和方法相衔接的对象管理系统 OMS,直接提供对象语义。

OMS 是所有关于对象信息的集中存贮和管理。OMS 用统一的方式管理所有对象,如:方便的服务引用机制和存贮机制。支持 OOA/D 的辅助工具等也作为 OMS 中的预定义构件来统一管理。

(2)对象型　OMS 对象的型规定了其所有实例对象的特点。所有对象共享共同祖先型 OBJECT 所拥有的最小共同属性和服务集。一部分对象型如图5所示,其中"构件"是持久对象的一个子型,其中预定义型包括原子型和复合型。工具看作是一种复合型预定义构件。

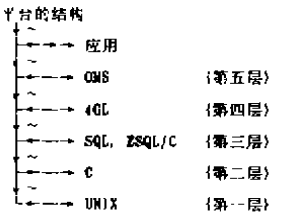


图4 平台的层次结构

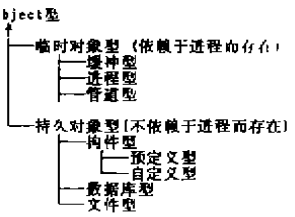


图5 对象型的组成

(3)OMS 的服务　OMS 所提供的服务如图6所示。

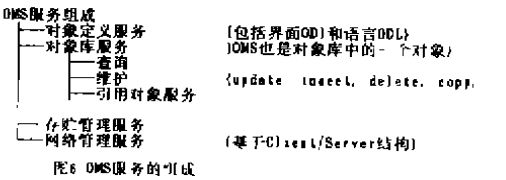


图6 OMS服务的组成

对象定义语言 ODL 是一个模式描述机制(附录1给出了对象定义语言 ODL 的描述(略)),它的解释器可以在模式定义集(SDS)之上负责将 ODL 语句转换为 SDS 上的服务。每个定义被创建时,都有一个系统范围内唯一的标识,根据命名规则将型定义组合到 SDS 中。

从 OO 的观点来看,MIS 可看作 OMS 或其它预定义 MIS 的一个子型。MIS 继承 OMS 或其它预定义 MIS 结构和服务的视图,不能通过继承 MIS 来得到的可通过继承预定义构件来实现,即,通过继承来构建新的 MIS。在传统方法中颇为棘手的软件结构设计的问题,在这里变得十分方便。

3.2　统一的虚拟对象机制

在 UNIX 之上建立 OO 系统需解决两方面的问题:一方面,进程对虚存的访问能力是很强的,但信息将随进程的结束而消失不具持久性,只有通过 I/O 系统将对象转换成字符流存入文件系统,但在将临时信息对象持久化的同时丢失了许多对象本身的语义信息。另一方面,文件系统具有持久性,但进程对字符流形式文件的操作能力很弱。因此,以统一灵活快速的方式来操作对象并使对象具有持久性成为 OMS 的一个核心问题。

OMS 层次结构如图7所示,在 UNIX 内核的基础之上将盘区空间(DS)和对象缓冲空间(TOM)都作为对象存贮空间不可分割的部分。为了解决 TOM 的空间限制问题,设计了提供虚拟对象存贮的机制,在虚拟对象空间(VOM)上,提供统一的对象服务。OMS 的所有服务都是针对 VOM 的,要访问的对象都是在一维空间 VOM 中的对象,在这一级上封装了临时和持久对象的概念,使存贮空间抽象化。

设 f 表示虚拟对象空间和实际对象空间的映射关系,f:VOM→TOM∪DS,对于任意对象 id∈VOM,如果存在 id′∈TOM,f(id)=id′,则 id 为活跃对象,如果存在 id′∈DS 且 id′不在 TOM 中,则 id 为睡眠对象。

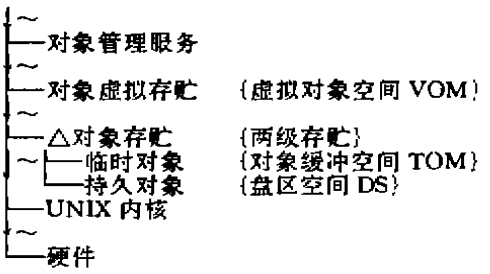


图7　OMS 层次结构

在物理实现上,为了提高效率,引入适当的存贮块对象 block(大小为1K),虚拟对象机制以 block 为基本单位进行读写操作。从存贮空间来看,DS 和 TOM 均由 block 大小的块组成。

对象表 O_TABLE 包括如下结构:

①对象索引:O_LIST(id,name,address,type,size,m_ptr,d_ptr,sub_ptr,sup_ptr,active,change,lock),其中 id 为对象标识;name 为对象名;address 为对象在虚拟对象空间中的地址;type 为对象类型;size 为对象大小(block 的整数倍,不足一个 block 的仍作为一个 block);m_ptr 为对象在 TOM 中的起始地址指针;d_ptr 为对象在 DS 中的起始地址;sub_ptr 为嵌入在对象 id 内所有成员对象组成的链表的头指针,当 sub_ptr=0表示无成员对象,其中每个结点都包含了成员对象 id 的信息;sup_ptr 为引用该对象的对象链表头指针(成员对象链和引用该对象的链也简单地可用固定个数的域来实现);active 为对象是否活跃的标记,当 active=1时,表示对象活跃,该对象已在 TOM 中无须重新调入,当 active=0时,表示对象睡眠,该对象已不在 TOM 中;change 为对象是否修改过的标记,如果对象未修改过(change=0)在睡眠时就无须再调出 TOM 而直接释放该块,以提高效率;lock 是锁标记,当 lock=1时表示该对象被锁,此时不能进行写访问,lock=0时表示该对象未锁。由于所有的操作都是针对 TOM

的，所以已在 TOM 中的对象对于后续操作都无需从 DS 再次调入，由此对象在 DS 中无须连续存放，但为了避免磁头移动过多，提高效率，相关的对象应分组连续存放在一起。

②临时对象工作空间(内存区)：名为 TOM，头地址 h_TOM，尾地址 t_TOM；

③TOM 的可利用空间块链：link_TOM，每个结点都包括如下内容：prt_node，post_node，可利用块头地址 head 和大小 size，在初始化时，可利用空间就是 TOM，该链在初始化时建立。

④持久对象盘区空间：名为 DS，头地址 h_DS，尾地址 t_DS。

⑤DS 的可利用空间块链：link_DS，每个结点都包括如下内容：pre_node，post_node，可利用块头指针 head 和大小 size，该链在初始化时建立。

这些结构存放在 DS 的头部，起始地址为 O_TABLE_begin，结束地址为 O_TABLE_end。

OMS 初始化的工作包括：①将 O_LIST 读入内存；②将一部分对象从 DS 装入 TOM；③建立 link_TOM 和 link_DS；④置每个对象的 change＝0。

实现虚拟对象机制由许多底层服务支持，下面是其中的调入和调出服务：

Load(id) /＊调入服务，从 DS 中将对象 id 读入 TOM，返回一个对象指针 m_ptr＊/

(1)根据 id 在 O_LIST 中找到相应的 size(id)，sub_ptr(id)；

(2)调用 find_m(size(id))找一个 link_TOM 中的可利用空间块 x，size(x)＞size(id)；如果 find_m(size)不成功则调用 unload(size(id))调出一个大于或等于 size(id)的对象；

(3)将地址 d_ptr 开始的对象读入 head(x)开始的块 x；

在 link_TOM 中删除空间 x；

置 m_ptr 为 x 的首地址；

(4)如果 sub_ptr 不为0，则调用 Load_sub(sub_ptr)依次调入其成员对象，置 active(id)＝1；

(5)返回 m_ptr；

(6)结束；

Remove(id) /＊调出一个完整对象，返回一个成功/失败标记＊/

(1)根据 id 找到相应的 size(id)，m_ptr(id)，sub_ptr(id)；

(2)调用 find_ds(size(id))分配一块合适的空间 x，置 lock(id)＝1，将 m_ptr 所指对象存入 x，置 lock(id)＝0，若无合适空间则调用紧缩服务 pack(link_DS)，若紧缩失败则返回失败标志；

将 m_ptr 所指空间释放给 link_TOM；

在 link_DS 中删除作 x；

置 active(id)＝0；

(3)如果 sub_ptr＝0，则调用 unload_sub(sub_ptr)，如果成功则返回成功标志否则返回失败标志；

(4)结束；

从语义角度来看，对象的语义与其 size 和地址无关，而与它及其成员对象的型密切相关。本系统将对象及其成员对象的型(包括原子型和复合型)作为对象强制语义解释的依据，并提供一套型的解释和管理机制。在初始化时 OMS 首先是将 O_LIST 从 DS 读入 TOM，然后再将一批对象读入。当虚拟机制置对象为活跃态时解释机制便根据对象及其成员对象的型做出某种语义解释。当需要置对象为睡眠态时对象及其成员对象的型也作为对象的一部分同时持久化，

另一个重要的问题是对象共享和保护，即进程访问对象受既定策略和规划的制约，本系统用行表示进程列表示对象一个访问矩阵[1]对资源进行管理。

四、结论

本文从实用观点提出一种 OO 分析设计方法，其优点是：(1)简洁，用该符号工具构成的图形没有交叉线，便于在有限的纸张上表现出更多的内容，给图的生成、打印、显示带来了方便；(2)分析阶段和设计阶段采用统一的一套符号系统，弥补了以往 SA/SD 分别采用两套符号体系及其自身图形符号并非结构化的不足；(3)表达能力强，可以表达多种关系结构，甚至还可用来书写用户使用说明书，从而使整个开发过程使用统一的一套符号系统。(4)与开发平台相衔接将使该方法贯穿于分析、设计、实现的全过程。分析设计文档及其辅助工具也作为对象来统一管理。

开发平台则以开放、标准、独立性和直接提供对象语义为目标，提供多层次的开发服务。通过我们近期开发的财政行业 MIS 群 CZ-MISs 和平台 MISDP 之实践表明，本文论述的方法和平台是可行的。进一步的工作包括：①提高平台的运行效率和独立性；②将其应用于企业 MIS 群的开发中，以期在实践中得到完善和发展。

参考文献

[1]Frank G. Soltis et al.，Design Consideration for the IBM System/38，in Object-Oriented Computing，Vol. 2：Implementations. G. E. Peterson (ED)，1987

(下转第30页)

统开发中好吗?结果并不乐观,往往长达几百页的数据流图,加工说明和数据字典令人望而生畏,可操作性实在是有些问题,O-O 方法到底怎么样呢?它对软件重用的支持到底有多大,效果如何,这些都有待于详细考察,从技术角度讲,这些是 CASE 生存和发展的关键因素。但不管怎么讲,当前 CASE 在我国已是破土而出,对于所有 CASE 领域的工作者而言,这是一个很好的机遇,应该抓住机遇,迎接挑战。

六、对下一步研究与发展的思考

我们认为应该首先展开对软件工程方法的研究,是否能找到一种比较合适的方法能真正适合于实际应用,而且能被广泛接受是我们软件工程研究者的当务之急。此外,我们认为未来的 CASE 环境应该尽可能地向人们认识规律和解决问题的实际方法靠拢,具体地说就是支持软件过程模型或者是某种高层次抽象的行为模型,并且引入对领域知识的表示和处理能力,使得用户感到 CASE 是他的有用的助手。智能化、集成化和开放性也许是下一代 CASE 的重要特征。软件重用技术、面向对象技术、领域知识表示和推理技术以及新型应用技术(多媒体,超文本,可视化……)也许是下一代 CASE 的核心技术,根据这些想法,我们感到下一代 CASE 也许具有图7所示的结构。

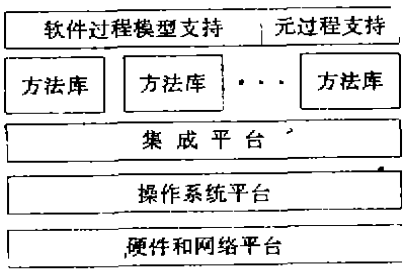


图7 新型 CASE 环境的构想

对 CASE 的研究应该把它的可用性放在第一位,这一点无论是对现在的 CASE 环境还是对未来的 CASE 环境都是一样成立的,CASE 的成败关键在于它是否真正适用,是否能被广泛接受。这一点已愈来愈受到 CASE 研究者和开发者的重视。K. Lyytinen 在谈到下一代 CASE 的特征时指出以下几点:

(1)改善了的可用性,

(2)在方法上的改进,

(3)加强对不同类型的表示的管理和它们之间的转换,

(4)强化协调多用户工作的能力,

(5)改善灵活性和适应性.

(6)加强 CASE 系统的分析能力。

在 CASE 的发展战略上,除了加强通用 CASE 平台的开发以外,应适时积极开拓专用 CASE 平台市场,值此国民经济信息化的大好时机,努力开发政府管理与决策部门、大中型企业、乡镇企业以及三资企业的应用软件及其开发平台的市场,以应用软件带动专用 CASE 平台的发展,以专用 CASE 平台推动通用 CASE 平台的研究、开发和商品化。

参考文献

[1]P. Mair,《The Strategic Procurement of CASE Tools》,NCC Blackwell Ltd,England,1993

[2]K. Lyytinen,《Next Generation CASE Tools》, IOS Press,Finland,1992

[3]M. R. Lowry, "Software Engineering in the Twenty-First Century", AI Magazine, Fall, 1992

[4]A. Fuggetta, "A Classification of CASE Technology", Computer, Vol. 26, No. 12, December 1993

[5]R. R. Huang,《Object Base Support for Process-driven CASE Environment》, Draft of Ph. D. Thesis, Department of Computer Science and Technology,Peking University,October 1994

(上接第42页)

[2]Hai ZhuGe and S. X. Hu, On Software Reconstrucion, AMSE, Hangzbou, China, 1991

[3]Hai ZhuGe and S. X. Hu, A High-Level Specification Language for the Development of Management System, AMSE, Hangzhou, China, 1991

[4]诸葛海 著,信息系统与类比方法论,浙江大学出版社,1992

[5]诸葛海,对象的类比推理研究,软件学报,1994,9:1-8

[6]诸葛海,面向对象的 MIS 开发方法,软件学报,1995,2:1-10

[7]诸葛海,面向对象概念和对象库的语义研究,计算机学报,送复审