第5卷第2期 软件学报 Vol. 5, No. 2
1994年2月 JOURNAL OF SOFTWARE Feb. 1994

集成CASE的集成模型*

麦中凡 张莉

(北京航空航天大学计算机系,北京 100083)

摘要 本文综述计算机辅助软件工程CASE概念的发展,从第一代文件系统集成的UNIX环境到90年代初的基于仓库的CASE环境.集成技术方面始终围绕数据集成、控制集成、表达集成发展.CASE的集成还要保证各厂家开发的工具和可重用成分可移植.为此,本文介绍了近代CASE环境集成的模式,CAIS和PCTE作为集成核心模型的原理,以及发展到80年代后期信息仓库的由来.信息仓库是PCTE成果的进一步发展,1988年ECMA提出以信息仓库为核心的烤面包实现集成模型.它是当今CASE环境研究开发的基本模型.本文详细论述了信息仓库的概念、原理、实现方法和问题,指出信息仓库是一新生事物,基于仓库的CASE环境对于软件开发具有革命性深远影响.最后,介绍国外基于仓库的CASE环境研究概况.

关键词 CASE,仓库,对象管理子系统OMS,数据集成,控制集成,表达集成,面向对象数据库.

长久以来,软件工程师为别的领域的自动化建立了大量复杂的软件系统,而为自身的自动化做得却很少.早在70年代中期,机械和电器工程师就将计算机软件用于该领域的分析和设计.随着终端的普及和图形工作站的出现,将大量支持设计制造的软件综合在一起就形成了计算机辅助设计CAD和计算机辅助制造CAM,后者进一步发展成为计算机集成制造CIM.它们统称为计算机辅助工程CAE.而软件工程自身倒晚一步,直到80年代初才开始计算机辅助软件工程CASE.CASE旨在以计算机软件辅助或代替软件工作人员完成软件工程的各项工作,最终实现软件生产、测试、验证、乃至维护的完全自动化.因而是大幅度提高软件生产率、保证软件产品高质量、降低软件开发和维护成本,实现软件工程目标的最有希望的手段.这些辅助软件工程的工具统称CASE工具,它们连同支持它们的硬件统称为CASE环境(CASEE).

毫无疑问,CASE给软件工程带来的影响决不亚于CAD/CIM给别的工程带来的影响.然而它们却基于完全不同的背景.CAD/CIM实现的是业经100多年实践所证明的工程理论,而软件工程技术却仍在飞跃发展:就软件开发模型而言,已从单一的瀑布模型转向快速原型、螺旋模型、第四代技术(4GT)等多种软件工程开发范型;而开发方法也从基于过程的各种结构化方法(HIPO、SA—SD、SADT、PSA/SPL、SREM)和面向数据的方法(JSD)发展

* 本文1993年5月28日收到,1993年9月17日定稿

作者**麦中凡**,59岁,教授,主要研究领域为软件工程环境,程序设计语言,数据库.**张莉**,女,26岁,博士生,主要研究领域为软件工程环境,软件生产自动化.

本文通讯联系人:麦中凡,北京 100083,北京航空航天大学计算机系

为面向对象方法、原型法、信息工程、形式化方法、集成化方法等多种方法.原本是为了有效地实施某种范型和方法而开发某些CASE工具,然而工具的出现又进一步推动了开发技术的发展.在这种相互作用下,CASE工具越来越多,这自然提出对工具的集中管理问题.其前题是如何将这些支持软件生存周期中不同阶段的工具,和横跨整个生存周期的工具,乃至横跨多个应用领域的各种模型方法和各种开发技术(AI、SE、OO、DB、网络)的工具等集成起来,并确保各个不同厂家的CASE工具间的协同工作已成为CASE研究的重点,从而集成技术也成为开发和维护CASE工具的关键技术.

1 集成CASE环境及其发展

所谓集成即用户在单一的界面上能控制与开发活动有关的所有工具与产物,使软件开发活动高效进行.

最初的集成就是以一个工具一个文件地"集中"起来,即文件集合的工具箱,其使用管理由系统程序员通过管道命令控制完成,并为用户提供统一的命令语言(Shell语言).这就是70年代初的第一代软件工程环境UNIX[9].它是现代集成CASE的雏形,因为它已包含了集成的三个主要特征.

·数据集成 从数据管理的角度,软件工程活动的各项产物(源程序及其各种文档),软件工具(包括工具的各种版本,私有数据、内码转换)都作为"数据"对象,一一分门别类放在库中统一管理.由于文件系统无语义,第一代CASE环境对"数据"的管理主要靠人,几乎没有什么自动.

·控制集成 "数据"管理可保持数据本身及其相关性的正确,如何保证使用正确,就是按某种开发模型正确地调配各工具及其使用的对象,使该种应用信息模型得以正确实施.正确调配在第一代CASE时是人们通过管道命令实施的,还没有"控制功能"的软件.

·表达集成 通过简单的界面,人们可以控制极其复杂的系统,正如司机有了方向盘、油门、刹车、指示灯就可以自如地驾驶汽车一样.第一代CASE环境只有比早期作业控制语言JCL(一般为单条命令)高级一些的Shell语言(正文方式).软件工程产物及各种文档也差不多都是格式正文文件.随着图形开发工具的引入(DFD,SC,PERT,PAD等)和图形技术的成熟(Window,OpenLook,Motif),图形表达和正文表达都能由集成的界面自如操纵.则表达集成自成为庞大的子系统,还要用到各种基础的图形标准(GUI,GKS).表达集成要解决各种信息表达的一致性和可交互操作性,为用户提供"视"、"觉"一致的界面.

在单机单操作系统上发展起来的第一代CASE环境确实改进了软件开发的生产率,各厂家纷纷效法在自己产品上建立CASE环境.工具与日俱增.70年代末就遇到新的问题.

首先,研制工具需要投入大量资金,而研制的CASE环境又是面向硬件系统的,因而提出了可移植性要求.其次,工具大量出现,而开发方法学又多种多样,工具作为"数据"有共享问题(不同方法学组合不同工具),软件工程产物的"数据"也有共享问题.而文件系统对文件持中性,有大量冗余信息,为了使频繁修改的开发活动能正常进行,就必须转向数据库的集成支持.消除冗余保持数据完整性.

于是,以Ada程序设计支持环境APSE[7,8,12]为代表的第二代CASE环境应运而生.

1984年美国国防部石人计划定义的APSE如图1所示.

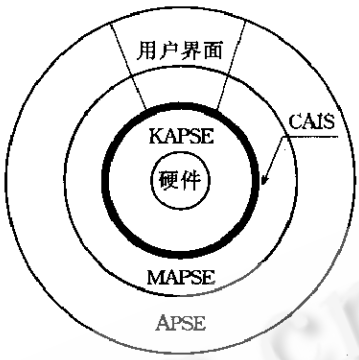

图1 APSE的体系结构

核环境KAPSE提供一通用的Ada程序设计接口集CAIS.在该环境外壳之上建立运行程序最必须的工具集MAPSE(最小Ada程序设计支持环境),包括编辑器、编译器、连接器、调试器、用户命令解释器、装载运行器等.

在MAPSE以上的外层是Ada程序设计支持环境APSE层,即一切有利于开发的软件工具.如格式美化器、静态、动态测试器、SC图——源代码自动转换器…石人计划并没有定义APSE的边界,它是开放的.

CAIS外壳既是抽象的操作系统,又是数据库的界面,集成在数据库中的程序、工具通过这个界面变为用户可操作的实体,而各操作系统按标准化的CAIS作出实现,则工具的移植就可以解决.

工具、程序、设备在CAIS中都抽象为文件实体.属性描述了它们的语义,关系描述了实体间的联系.文件的执行由初始化进程结点按文件的静态属性和关系生成一个实例进程,以及相应出现的动态关系(关系实例)和派生性模型,即由进程、文件、结构三种实体之间关系(主、辅)组成的有向图.文件变为结构(有属性、关系描述),结构变为进程.具体机器按此模型实现进程管理,即可实现工具、程序、数据的集成使用.美国国防部1984年—1986年投入大量研究,1986年形成DOD—STD—1838标准.

与此同时,西欧的软件工作者提出以数据库为集成核心的IPSE(集成的程序设计支持环境)[7],其体系结构大体和APSE相似,只是不依赖单一的程序设计语言.这样,对它的可移植性和可交互操作性要求更高.为此,欧洲共同体ESPRIT计划首先开发"可移植通用工具环境PCTE标准".并且PCTE支持分布式应用.PCTE实现原理和CAIS完全一样,提供一个抽象的实体(工具、程序、数据、设备)之间的关系图(实体间固有的静/动态联系).因而1985年最早研制时,采用C语言和扩展的E—R模型.只要新开发的工具符合PCTE标准,按其要求的属性关系描述,就可以加入到可移植环境中.

80年代面向对象技术蓬勃发展,PCTE的研制者很快地转向OO数据模型,并定义了一个对象管理子系统OMS.将软件工程环境下的各程序、工具、数据、设备统一作为对象进行管理,其数据和特征描述为对象的属性;相应的操作即对象的方法;工具的激活(执行)和数据传递对应为对象间的通信;数据和操作的共享对应为继承.由于当时尚未出现商用OO数据库,因而它仍是基于RDB的.当然,OMS功能不止于描述各对象静态关系和充作存储模型,也是动态执行相关性的模型(进程、关系(链接)都是对象).1987年欧洲计算机制造商协会ECMA提出了建立在对象模型上的PCTE+.在此基础上,1988年完成了PCTE149标准.它实际上是集成模型框架和开发平台.

80年代后期出现了基于信息仓库(Repository)的CASE环境,以1988年IBM公司推出的系统应用体系结构(SAA)为代表.它提供整个企业范围内的信息仓库,该仓库集成了

一套用于规划、分析、设计、编程、测试、维护的工具. 程序员在 SAA 的 AD/Cycle 平台上作应用项目的开发. 他只需根据业务目标选取信息系统(IS)的部分结构和成分,拼凑成应用项目,然后测试、文档、交付. 最终用户在 AD/Cycle 平台上可以直接访问 SAA,了解本项目的目标、结构与功能,所作的处理,数据流程,业务信息以及它们的关系,查找能满足本项目已有的应用系统. 从而能及早参与开发,甚至不用参与开发立即进入使用培训. SAA 的出现把开发的原型化和软件重用推到一个新的高度. 应用开发从项目为中心转向面向处理过程. 这是一种革命性的转变,其技术基础是现代集成 CASE 环境的信息仓库技术.

1983 年英国 Alvey 计划的一项策略报告[8]中曾以文件库、数据库、知识库为集成核心把 CASE 环境划分为三代,并预言 90 年代会出现第三代 CASE 环境. 虽然直到目前并未出现第三代 CASE. 然而,把人工智能技术引入 CASE 环境,部分成果商品化已成为现实,例如,按规则推演版本(DSEE 中),以知识件辅助项目规划(SAA 中). 随着第二代 CASE 引入面向对象技术,结合人工智能将更加方便(对象模型和知识表示模型的相似性). 问题仍是效率.

总的说来,CASE 环境技术仍处于第二代,但以数据库为集成核心的概念经多年实践变为以信息仓库为核心. 信息仓库系统是一种新技术,它不仅是 CASE 环境的专用数据库,而且对数据库技术本身也会有重大影响. 后文还要介绍. 为此先谈谈近代 CASE 环境的总体结构.

2 现代集成 CASE 的体系结构

现代 CASE 环境一般建立在网络、工作站、多用户操作系统之上. 从概念上可以看成如图 2 的构件块组合[3]. 最下层为硬件体系结构. 基于网络的操作系统建立在该体系结构的硬件平台上. 第 4、5 层为可移植设施和集成框架. 前述 CAIS 和 PCTE 提供的界面模型即为可移植基础. 在此模型上建立的各种管理系统即构成集成框架,如,仓库管理、进程管理、消息服务器、数据集成管理、用户界面管理. 从功能上实现数据、控制、表达集成.

最上层是 CASE 工具集和信息系统 IS 模型. 通过开放式、灵活的集成框架,按不同 IS 模型提供经济的工具集成机制. 使 IS 的结构能在开放的系统平台间及时传送和转移.

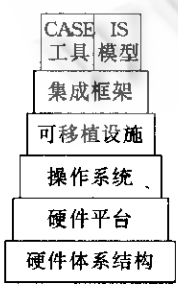

图2 集成"构件块"

1988 年欧洲计算机制造商协会 ECMA 将图 2 的第 4,5,6 层具体化为"烤面包"的实现模型,如图 3. 它是 APSE"洋葱头"模型的深化. 继承了它的开放式的思想,即新工具以"软插件"的方式插入,数据集成设施将它描述为"元数据",再存入仓库.

用户界面设施 提供用户以图形、正文、图标方式表达开发信息的表达集成. 提供应用开发(程序员)和(最终)用户访问的友好工作界面.

进程管理设施 依据信息模型,将库中的工具和仓库中软件产物按开发工作流程组织为所需的进程,激活、调度、派生、执行、删除各进程.

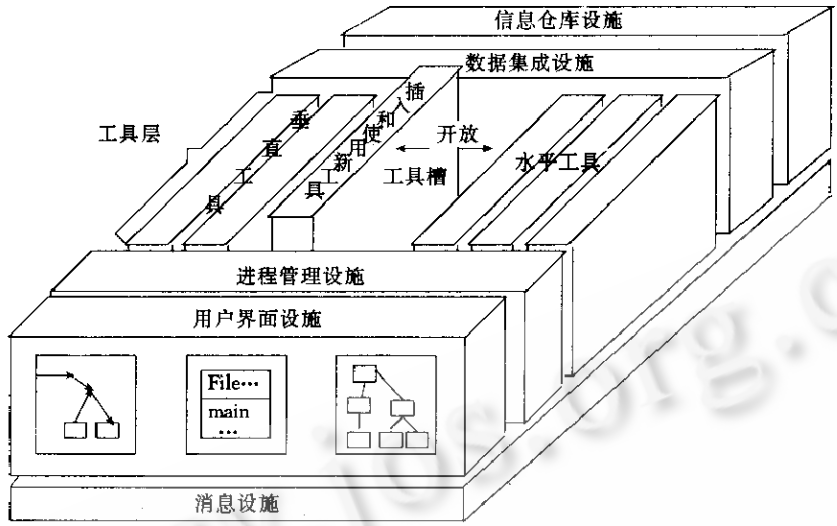

图3 NIST/ECMA烤面包模型

消息服务器　是操作系统和本模型环境的纽带,实现进程执行和整个工作流程的协调,它与进程管理设施一起构成控制集成,即按有关方法学指导协调使用工具.

数据集成设施　将软件工程活动的各项产物和工具加以元数据描述(对象化)存入信息仓库.

信息仓库　(Repository 也译仓库、中心库、项目数据库、信息仓)是集成核心.它是CASE 专用的信息库,一般数据库系统(RDBMS,OODBMS)只是实现它的成分.简单地说:

信息仓库=信息模型+控制功能+DBMS

仓库按企业的某种信息模型组织、存放、使用各种工具与信息,按信息模型生产、使用、维护各项目应用.应用开发的各个阶段都直接和仓库打交道(通信、传播都要通过仓库),它分析、维护各软件项的依赖关系;调配使用各种存储系统(DBMS,文件系统).以达到充分的数据共享和高度的集成.控制功能包括各软件项的版本控制、工作流控制、配置控制,以及软件项(广义数据)出入仓库的检定(Check—in,Check—out)控制,实现正确配置,可对更改与版本作传播控制,并有通知设施报告追踪信息.

IBM 的 SAA 系统的集成框架和软件平台 AD/Cycle 的实现模型即本模型实例,如图 4 所示.其中工作站设施和工作项管理即进程管理设施的具体化.工具设施和 AD 信息模型即数据集成设施的细化.增加库是为了提高效率.对于更改不频繁的少量、重要系统程序放于库(Library)中.

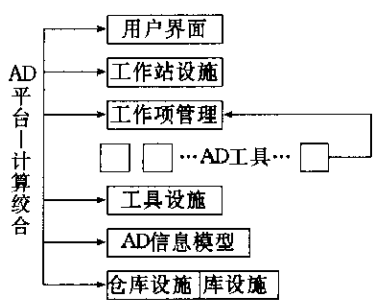

图4 AD/Cycle体系结构

3 信息仓库

仓库是文档、转移、维护信息系统信息的软件.是数据共享、集成设施.为控制软件更改的影响和传播,为程序员、设计员、经理人、开发工具提供全生存期服务.软件开发活动各阶段都直接和它打交道,软件工程活动产物(项)的通信和转移都要通过它.为此,仓库要作依赖性维护,管理软件项存入存储系统(数据库、库).

显然,仓库不同于一般数据库,它虽然也要按一定模式存放"数据",但模式是连续演进的,且直接由多个用户组更新而不是由DBA.它的数据类型丰富,有格式、无格式、对象、规则等,而不象DBMS只许格式记录.类型多(不同语言、不同阶段)、实例少(往往一个或几个版本).DBMS则相反,类型少实例数据多.仓库初始化时数据少,开发期间骤然增多,以后增长缓慢;数据库一开始数据就要很多以后稳定持续增长.仓库更新数据是间接的而不象DBMS直接通过SQL完成.更重要的是仓库中的事务一般是长的非原子事务,例如,一个用户的开发阶段可持续几天甚至更长;DBMS中是短小原子事务.

仓库也不同于库(Library),库只能按关键字查找;也不同于数据字典,后者只提供存储、访问模式,无模式演进和维护;也不同于主动(Active)数据库,后者的触发机制有利于维护一致性.但仓库集这些技术之大成.利用OODB的分类、概括、聚集机制,建立对象字典,按信息模型建立具有触发机制的对象管理系统(OMS),提供可实施动态模式演进、更改追踪、访问保护和安全的基础,显然OODB是实现仓库最有希望的候选者.

信息仓库管理器要实现跨生存期集成,就必须按信息模型组织各软件项和工具,众所周知,软件开发活动中还没有适应全生存期的模型,为建立它们的统一管理则要进一步抽象为"统一"的信息模型,也就是模型的模式(表达子模型相关性).为支持模式扩充和演进再进一步抽象为元模式.以后每加入一种新模型即为元模式的新实例.为此,美国国家标准局1988年制定信息资源字典系统IRDS(ANSIX3.138)标准,它将信息资源分为四个层次,见表1.

表1 信息资源分层表

第4层	IRD模式描述层 元模式
第3层	IRD模式层 信息模型
第2层	IRD数据层 系统成分模型
第1层	产品数据层 数据

信息仓库的分层管理,其元模式可采用扩展的实体—关系模型和OO对象模型等.信息模型,如IBMSAA的企业模型.各阶段模型,如系统分析、系统设计,代码自动生成、项目管理、版本与配置管理等均为子系统成分模型.

但是IBMSAA的企业模型仅适于数据处理领域的业务.如何扩充到更广阔的领域还是研究的课题.此外信息模型要独立于表达语言,且各种表示法(格式、无格式、对象、约束、演

译规则、多介质)都要能接受,仍然有许多课题.第三,现时尚未找到更改软件项特别是成组用户更改的模型,因而依赖性管理只能就事论事,个别解决.第四,随着可重用成分、工具日益增多,庞大的仓库如何建立在异质分布式网络各结点上又能协调一致地工作,如何利用并行计算减少查找、推理时间,都是要解决的实现问题.

4　当前基于仓库 CASE 环境的有代表性研究

目前,在基于仓库的集成 CASE 环境方面较有代表性的研究成果有:美国政府(主要是 DARPA——the Defense Advanced Research Project Agency)支持的 Star、Arcadia 和 Prototech. Star 项目支持工具的高级集成、超前软件过程的开发,并支持软件构件和过程的重用;Arcadia 是 Ada 的开发环境,包括环境体系结构、软件过程、对象管理、用户接口和测试与评价五个方面的研究.其对象管理系统 Cactis DBMS 具有较大的灵活性和可扩充性,且有依赖性管理. Prototech 项目侧重于对快速原型法的支持.它支持多语言、多种原型开发方法;提供了较强的测试,评价,以及先进的智能编译功能,此外为了集成环境构件,还提供了模块接口设施 MIF(Module Interface Facility).

为了集成现有的商用工具,HP 公司的 SoftBech 作出了新的尝试,在对象管理的基础上,其信息传送用了广播传送的方式,优点在于使得已有工具易于封装,并能发送和接收信息,工具封装器使得已有工具无需较多修改便可集成放入环境中.

此外,作为仓库系统,DEC 公司于 1988 年也推出了 COHESION—CDD/Repository 和 CDD/Plus 系统.其软件仓库模型是一个分类层次结构.除此之外,关于软件仓库的产品还有:MAP 数据管理器、SOFTLAB 的 Maestro Ⅱ 仓库、图形制表软件的 IDE 仓库,以及仓库和 CASE 环境交互操作的 CDIF 标准等.

结束语:基于仓库的 CASE 环境的基本目标是软件开发从规格说明到设计到代码生成全部自动化,软件维护修改半自动或自动.这必然要引入基于规则的描述,加上大量可重用软件成分(对象).管理配置这些可重用成分要借助信息模式和引入推理,以此维护仓库中集成数据的完整性.此外,现代软件开发强调用户参与.因此基于规则的描述不得不套上一层用户易理解的图形工作界面.这就要求表达集成.再者,为了支持原型(或螺线型、4GT)开发,各工具进程的工作流和加工对象(软件项)配置控制要求控制集成.这样,仓库管理器就构成三种集成的核心,是支持现代软件开发最有希望的技术.

基于仓库的 CASE 环境还刚刚开始.各厂家推出的环境大致符合烤面包模型.但马上面临全球信息网挑战.各可重用成分、各工具、开发方法如何在世界范围内标准化.这就涉及更深一层计算机文化与各国文化协调的问题了.

参考文献

1　Ronald J. Norman, Minder Chen. Working together to integrate CASE. IEEE Software, 1992;3:13—16.

2　Minder Chen, Ronald J. Norman. A framework for integrated CASE. IEEE Software. 1992;3:18—22.

3　Roger S. Pressman. Software engneering. McGraw—Hill, Inc., 1992.

4　Product Catalog. Interactive development environments, 1992.

5 Thomas Rose, John Mylopoulos. Software repositories. Tutorial Notes from VLDB'92, August 23－27, 1992, Vancouver, Canada.
6 Software development environments research projects in the United States. Leon Osterweil.
7 麦中凡,李昭智. 集成软件项目支持环境 IPSE. 软件产业,1989. 9.
8 周伯生,董士海. 软件工程环境引论. 计算机研究和发展,1986. 7.
9 BW, Kermighhan. The UNIX programming environment. Prentice Hall, Inc., 1984.
10 Stephen L Montgomery. AD/Cycle: IBM'S Framework for application developement, 1991, IBM.
11 IAN Thomas, Hewlett－Packard, CASE Brain A Nejmei. Definitions of integration for environments.
12 Robert Munck, Patricia Oberndorf, Erhard Ploedereder. An overview of DOD－STD－1838A, The Common APSE Interface Set, Revision A.

INTEGRATED CASE TOOLS AND REPOSITORY

Mai Zhongfan and Zhang Li

(*Department of Computer Science, Beijing University of Aeronautics and Anstronautics, Beijing* 100083)

Abstract This paper summarises the development of the concept of Computer Aided Software Engineering(CASE) from the first generation file－based integrated UNIX environment to the repository－based CASE environment in the early 1990s around data integration, control integration and presentation integration. So it introduces the integrated model of current CASE environment, the principle of the model with CAIS and PCTE as its integrated kernel and the origin of the repository in the late 1980s. Repository resulted from the further developnment of PCTE and in 1988 ECMA presented repository－based Roast Bread accomplishment integration model which is the basic model in the current research of CASE environment. Then it discribes the concept, principle, accomplishment methods and problems in repository in detail and points out that the repository－based CASE environment will have deeply influence on software development. At last, the state of the art of the research in repository－based CASE environment is briefly introduced.

Key words CASE, repository, object management system(OMS), data integration, control integration, presentation integration, object oriented data base.