

# 基于 UML 的集成化 CASE 平台的研究和实现

方木云,戴小平

(安徽工业大学 计算机学院,安徽 马鞍山 243002)

**摘要:**统一建模语言提供了研制和开发面向对象软件集成化开发环境的理论基础,指出 UML 本质上是一套文档符号,提出 UML 增减原则:是否有利于自动文档生成;是否有利于自动程序生成。提出一个“开环”的集成化开发平台的构造框架。最后,基于上述设计思想和结合软件项目实际情况,开发出一个集成化 CASE 平台,平台具有数据库文档自动生成、界面自动生成、代码自动生成和构件重用等功能。

**关键词:**UML;面向对象;集成化 CASE 平台

**中图分类号:**TP311

**文献标识码:**A

**文章编号:**1005-3751(2006)02-0026-03

## Research and Realization of UML - Based Integrated CASE Environment

FANG Mu-yun, DAI Xiao-ping

(Dept. of Computer Science, Anhui University of Technology, Ma'anshan 243002, China)

**Abstract:** Unified modeling language provides the foundation of constructing integrated object-oriented software development environment. UML is a set of document diagram. The principles to develop UML are: whether it is useful to produce document automatic; whether it is useful to produce program automatic. The frame of an unclosed UML-based integrated CASE environment is proposed. In the end according to the above discussions and the fact case of software project, an UML-based integrated CASE environment with functions: forming database files; forming interface; forming code; components reusing, is developed.

**Key words:** UML; object-oriented; integrated CASE environment

### 0 引言

1989~1994年期间,面向对象的分析与设计方法和面向对象的建模语言数量达到50种以上。其中最有力的是G. Booch, J. Rumbaugh和I. Jacobson三位方法学家分别提出的Booch方法、OMT方法和OOSE方法。这些方法晦涩难懂,相互之间符号混杂,使得整个面向对象领域的学习、教学和应用非常困难。为了改进各自的方法和便于走向市场,1994~1995年,G. Booch, J. Rumbaugh和I. Jacobson开始将3种方法统一起来,形成统一建模语言UML,于1996年6月和10月分别发布了二义性较少的UML0.9和UML0.91版本。1997年11月,UML正式被对象管理组织OMG采用<sup>[1,2]</sup>。

近几年国内也开始了UML的应用和研究:应用上,UML和支持UML的CASE工具Rational Rose在许多软件公司和高校的教学中被采用。研究上,北京大学的邵维忠等学者一直本着独立思考、实事求是的态度来对待

UML,对UML的现存问题和发展道路作了研究,作出许多客观、科学的评价<sup>[2,3]</sup>;北京航空航天大学的周伯生等学者对基于UML的集成化软件开发环境作了不少研究工作<sup>[4]</sup>;北京理工大学的陈英等学者对基于UML软件开发过程的CASE平台进行了研究<sup>[5]</sup>;国防科技大学的齐志昌等学者对UML statecharts的测试用例生成方法作了研究<sup>[6]</sup>。这些研究工作针对性很强,而且卓有成效。

UML已经成为软件工程领域近10年来最重要的研究成果,但UML并不成熟,正在发展之中,研究方向有:精简UML、定义UML核心、定义UML扩展机制、UML形式化语义定义和UML集成化软件开发环境的研究,其中基于UML的集成化软件开发环境的研究应该是最重要的方向,其它方向是这个大方向下的子问题。

文中分析了UML建模语言的本质和优缺点,然后在此基础上提出集成化软件开发平台的设计框架,最后结合多年ERP软件开发的经验,开发了该平台的一个实例。

### 1 UML是一套描述能力强的图形化文档符号

UML传到国内之后,一部分人对其推崇备至,容不得别人对UML的怀疑;一部分人保持清醒的头脑,对其有褒有贬,北京大学的邵维忠等学者就是这方面的典型代

收稿日期:2005-05-08

基金项目:安徽省教育厅资助项目(2005KJ076)

作者简介:方木云(1968—),男,湖北罗田人,副教授,研究领域为软件工程、软件质量度量。

表;大部分人对其一知半解,感觉它太难理解,保持一种学习和观望的状态。那么,UML 到底是什么? 尽管 G. Booch, J. Rumbaugh 和 I. Jacobson 三位学者各自提出了面向对象的方法,但他们合作产生的 UML 却不是方法。邵维忠将 UML 总结为:UML 不是建模方法,不是编程语言,不是形式化语言,它只是一种独立于过程的可扩充的建模语言<sup>[2]</sup>。文中的观点是:UML 是用来进行需求分析、设计和实施的一套统一的可视化图形符号,本质上是一套软件开发的文档符号。

UML 的优点是:直接面向软件全过程建模,因此不存在过程转换中的信息丢失;图形符号多,从时空立体角度来描述一个系统,因此表达能力非常强;在 Rational Rose 等工具的支持下,能自动生成一些代码,这点非常吸引人,因为让人看到软件产业自动化的曙光。

UML 的缺点是:太复杂、难以掌握。因此,UML 面临精简和定义核心的问题,以便于学术界、工业界和教育界更容易接受它,但它的精简并不只减不增,应该有减有增,那么增减的原则是什么呢? 总原则是:一切有利于自动软件生成。由于软件由程序和文档组成,这个总原则又可以分解为两个小原则:有利于自动程序生成;有利于自动文档生成。由于现在的软件往往是可视化的,所以自动程序生成原则又可以分解为:有利于程序界面的生成;有利于程序代码的生成。下面将这些原则用图形表示如图 1 所示。

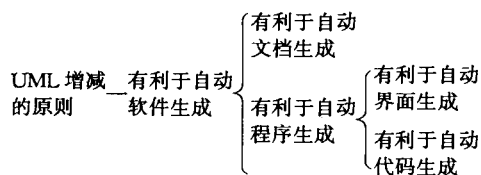


图 1 UML 增减的原则

按照这种原则,UML 的有些图形需要合并,如:顺序图和协作图可以合并,原因是信息冗余;而模块结构图和程序流程图的思想可以容纳进来,有利于自动界面和自动代码生成。软件开发经历了个人、作坊和工程化 3 个阶段,现在逐步进入自动化阶段。而软件开发的自动化依赖于集成化开发平台的支持,下面讨论集成化开发平台的构架。

## 2 基于 UML 的集成化 CASE 平台的构成

软件规模和复杂程度在不断增加,用户对软件开发的周期要求越来越短,质量要求越来越高,同时软件公司逐年增多,市场的竞争愈来愈激烈。现有的开发方法跟不上社会对软件的需求,人们提出了面向对象的方法、构件复用等思想,但这些都没有最大程度上提高软件开发的效率和质量。支持自动软件生成的集成化开发环境也许才是真正解决这一问题的途径。不少文献已研究集成化开发环境的构成<sup>[4,5]</sup>,这些文献共同观点是:环境支持从上游到下游的正向转化和从下游到上游的逆向转换,是一个

“闭环”开发环境,保证代码和文档的一致。这种思想理论上比较完备,实现起来应该比较难,比如,如果文档向代码转换,转换的粒度是多大? 多画一张图,代码改变多少? 多画一条线,代码要改动吗? 所以上游文档向下游代码的转换实现比较困难。如果代码向文档转换,转换的粒度又是多大? 增加一条冗余语句,文档需要变化吗? 改变变量的名称,换一种操作符,文档需要变化吗? 所以逆向转换在实现上也比较难。

如果将软件项目看成一元方程,它必有一个精确解,这个精确解首先驻留在系统分析员的思维里,这里称其为思维解;然后被翻译成软件文档,这里称其为文档解;开发人员将这个文档翻译成代码,这里称其为代码解。传统的软件开发就是按照这种模式来进行的,文献[4]和文献[5]提出的集成化开发环境在传统模式下增加文档解和代码解之间的相互转换。这两种模式都存在共同问题,文档解和代码解都是思维解的表现形式,它们之间的转换必然带来对思维解的误差,因此,它们都应该直接来源于思维解。文档解主要面向用户,代码解主要面向开发人员,文中的集成化开发平台就是基于这种思想来构造的。下面是传统软件开发步骤、文献[4]和文献[5]的软件开发步骤、本文的软件开发步骤比较,如图 2 所示。

软件项目  $\xrightarrow{\text{系统分析员}}$  思维解  $\xrightarrow{\text{系统分析员}}$  文档解  $\xrightarrow{\text{程序员}}$  代码解

(a) 传统软件开发步骤

软件项目  $\xrightarrow{\text{系统分析员}}$  思维解  $\xrightarrow{\text{系统分析员}}$  文档解  $\xrightarrow{\text{程序员+UML 集成环境}}$  代码解

(b) 文献[4],[5]的软件开发步骤

软件项目  $\xrightarrow{\text{系统分析员}}$  思维解  $\xrightarrow{\text{系统分析员+程序员}}$  UML 集成化  $\rightarrow$  文档解  
CASE 平台  $\rightarrow$  代码解

(c) 本文软件开发步骤

图 2 开发步骤比较

从图 2 比较来看,文中所设计的 CASE 平台应该支持从中游向上游和下游两个方向同时转换,集成化环境提供一个公有平台供编辑,然后向上游的文档和下游的程序自动转换。从逻辑上讲,这是合理的,既然追求文档和程序的自动生成,就不应该去手工写文档和代码,因此不存在正、逆转换。系统分析和软件开发人员只编辑中游的一个设计环境,然后随时向两边刷新,这种转换显然是一致的、快速的、安全可靠的。这种思路可能使问题变得简单,集成化平台设计者只需要将主要精力放在中游的公有平台设计上。文档解和代码解是思维解的两种输出方式,是一致的,但表现形式可能是丰富多彩的,而中游的思维解是唯一的,表现形式应该单一和严谨,是集成化开发环境的驱动力。文中的集成化开发环境是从中游向上游和下游同时转换的“开环”环境。

图 3 阐述了基于 UML 的集成化开发环境应具有的功能。

### 2.1 软件文档生成系统

软件文档包括从系统分析到维护的所有文档,在该平台上都应该以项目编号来进行组织和管理。

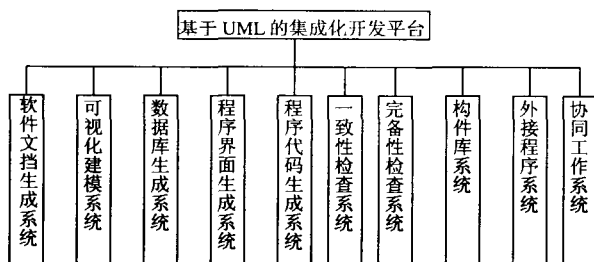


图 3 基于 UML 的集成化开发环境结构图

## 2.2 UML 可视化建模系统

主要是 UML 的所有视图的绘制系统。

## 2.3 数据库生成系统

类是面向对象软件开发的核心,数据库的表对应 UML 类,一个类对应一张表,类的属性对应表的字段,类之间的关系对应表之间的约束关系,所有面向对象软件都存在数据库建设的问题。

## 2.4 程序界面生成系统

程序界面由用例图和类图来共同生成,一个用例往往对应一个上层模块,类的一个方法对应下层模块。增加模块结构图主要是为了将用例和类的方法进行综合整理,以便更容易向界面转换。

## 2.5 程序代码生成系统

针对界面的一个菜单或一个按钮的某个事件生成其相应代码。由于面向对象的程序是事件驱动的,所以代码必须针对对象的某个事件来生成。

## 2.6 UML 一致性检查系统

UML 有严格的语义定义,为保证模型符合语义规定,系统中应该提供语义正确性检查机制,靠开发者手工来维护模型间的一致性是不现实的。

## 2.7 UML 完备性检查系统

UML 语言为了保证自动软件生成,语言体系是完备的,对一个软件系统的描述也应该是完备的,因此集成化开发环境应该提供完备性检查机制。

## 2.8 UML 构件库系统

一些通用的构件可以存储在构件库内,供程序重用。

## 2.9 UML 外接程序系统

主要考虑与 Word, Excel 和 PowerPoint 等应用程序的连接,供文档管理和演示等用途。

## 2.10 协同工作系统

一个大的软件系统需要许多人来开发,因此,集成开

发环境必须能支持多人的协同工作。尽管自动软件生成系统开发出来以后,开发人员会减少,但提供多人协同工作的环境还是必须的。

## 3 基于 UML 的集成化 CASE 平台的实现

基于上述 CASE 平台的设计思想,结合 ERP (Enterprise Resource Planning) 软件的特点,以 VB6.0 作为开发工具,SQL Server2000 作为数据库,构造了一个实用的集成化 CASE 工具,特别适用于 ERP 软件的辅助设计。

该 CASE 工具已实现的功能有:

1) 文档管理:调用 Word 来进行合同、进度和人员的管理,主线是项目编号。

2) 自动数据库和数据库文档的生成:当类及其属性定义好后,通过 CASE 工具自动建立好数据库并形成整个数据库文档。

3) 自动代码生成:以数据库的表为中心,对该表的保存、更新、删除、查询等操作的代码自动生成。

4) 自动界面生成:以数据库的表为中心,对该表的保存、更新、删除、查询等操作的界面自动生成。

“中游”在该 CASE 工具中体现为等待用户输入的界面,如图 4 所示的是 VB 代码生成的界面。



图 4 VB 代码自动生成界面

从图 4 中可以看出,每选择数据库中的一个表,针对该表的 4 种操作的 VB 代码自动生成。该 CASE 工具创建的数据库可以是 SQL Server, Oracle, Access 等,并生成其数据库文档;可以生成 VB, Delphi, PB, C# 和 Java 等开发工具的主要界面和核心代码。在课题组和一些软件公司应用效果非常好,提高了软件开发效率,降低了代码出错率,同时对软件公司的代码起到很好的规范作用。

## 4 进一步的研究工作

该平台已经实现了核心代码、核心界面、数据库、数据库文档的自动生成和构件库的重用等功能,后面的研究工

(下转第 31 页)

程序负责对样本烟叶进行图像采集,并通过网络把图像发至控制室中的服务器端。当服务器端程序收到图像后,便会自动调用 Matlab 程序对其进行大片率分析。

文中使用的叶片图像分析算法<sup>[4]</sup>如下:

1)图像除噪。此环节使用函数 `wiener2()` 进行自适应的维纳滤波。

2)叶片轮廓提取。此环节分3步进行:

A、使用函数 `edge()` 进行边缘检测。

B、考察检测到的边缘的邻域,以此邻域是否包含背景为依据,使用自编函数去除由粗大脉络生成的虚假边缘。

C、使用函数 `imdilate()` 实行膨胀操作,连接叶片轮廓的间断点。

3)填充轮廓。此环节分两步进行:

A、使用函数 `imfill()` 进行轮廓填充,获得背景和烟叶完全分离的二值图像。

B、使用函数 `imopen()` 进行开运算,扫除细小碎片的干扰。

4)叶片面积统计。此环节分两步进行:

A、使用函数 `bwlabel()`,确定每个叶片所占据的图像区域,并分别对其进行标识。

B、使用函数 `imhist()` 统计每个叶片所占据像素的数目,获得面积。

需要指出,以上所列函数均来自 Matlab 的图像处理工具箱,这使得此算法的实现异常简洁。以第2步的A操作“边缘检测”为例,可以在 C++ 中直接做如下的调用:

```
engEvalString(ep, "image = edge(image, 'canny');");
其中“ep”为此前建立的 Matlab 引擎指针,而引号内的则为将要执行的 Matlab 指令,其中字符串‘canny’表明使用 canny 算子进行边缘检测[5]。
```

表1 叶片分布统计表

叶片面积(像素)	2000以下	2000~3000	3000~4000	4000~5000	5000~6000
数目:	0	1	3	1	3
叶片面积(像素)	6000~7000	7000~8000	8000~9000	9000以上	/
数目:	1	5	1	2	/

(上接第28页)

作集中在:“中游”公共编辑平台的进一步建立,现在的窗口有5个:建立数据库;生成界面;生成代码;文字性文档管理;构件库重用。需要增加向图形转换的窗口。另外类之间复杂调用关系的代码生成能力不够,必须增加功能更加强大的编辑窗口。下一步准备用 C# 来实现该平台,增强图形绘制功能。相信在未来的几年内,该平台将被完善成一个强大的软件自动生成工具,成为 ERP 软件开发的重要利器。

#### 参考文献:

[1] 王萍,蔡雨阳,黄丽华.面向对象建模方法——上海卷烟

如图1所示的烟叶图像,使用上述算法,可求出叶片面积的分布,如表1所示。

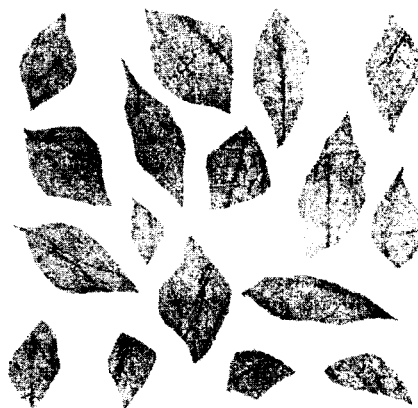


图1 叶片图像(512×512,缩小2倍显示)

#### 4 结束语

文中所讨论的混合语言开发是一种各取所长的技术: Matlab 中许多耗时或者无法完成的工作对于 C++ 来说是轻而易举的,而某些 C++ 非常难以实现的算法对于 Matlab 又是易于实现的。分布式技术又使得一台服务器上的 Matlab 可以被多个客户端共享,既有效地利用了资源,又降低了解决方案的复杂性。

#### 参考文献:

- [1] 徐科,杨朝霖,李滨涛. C++ Builder 实用技术与经典案例[M]. 北京:清华大学出版社,2002. 283-303.
- [2] 李幼仪,甘志. C++ Builder 高级应用开发指南[M]. 北京:清华大学出版社,2002. 395-401.
- [3] MATLAB, External Interfaces, User's Guide, Version 6 [Z]. 2002.
- [4] MATLAB, Image Processing Toolbox, User's Guide, Version 3.2 [Z]. 2002.
- [5] Castleman K R. 数字图像处理[M]. 朱志刚等译. 北京:电子工业出版社,2002.

厂计调系统分析与设计[J]. 中国管理科学,2000,8(S):459-466.

- [2] 邵维忠,杨美清. 面向对象的系统设计[M]. 北京:清华大学出版社,2003.
- [3] 邵维忠,蒋严冰,麻志毅. UML 现存的问题和发展道路[J]. 计算机研究与发展,2003,40(4):509-516.
- [4] 田丽从,张莉,周伯生. 基于 UML 的集成化软件开发环境的研究与实现[J]. 北京航空航天大学学报,2003,29(10):935-938.
- [5] 陈英,施兴华,赵小林. 基于 UML 软件开发过程的 CASE 平台研制[J]. 北京理工大学学报,2001,21(1):49-51.
- [6] 李留英,王戟,齐志昌. UML statecharts 的测试用例生成方法[J]. 计算机研究与发展,2001,38(6):691-697.