面向对象的软件结构设计及其辅助工具

王黎明

（郑州大学计算机科学系）

摘要：本文主要讨论面向对象的软件结构设计过程和方法，然后介绍辅助软件结构设计的一个工具及一些实现技术和特点。

自从出现软件危机以来，相继提出了不少的软件开发方法和分析技术，如、结构化设计方法，JSD设计方法，HOS设计方法，这几种方法都是目前比较典型的分析设计方法。结构化设计方法是直接从描述问题的数据流程图转换得到求解空间中的软件结构，但这种方法的分解过程主要强调功能的分解，而不注重数据方面的抽象，这样就会导致求解空间和问题空间在结构上的不一致性。JSD方法是在JSP方法的基础上建立起来的[3]，它是直接从现实世界中识别实体、活动及它们之间的关系，然后分析实体的结构和实体之间的关系，导出软件的结构，但这种方法对数据结构的依赖性太强，不可避免地要导致这种方法对问题求解的局限性。HOS方法是一个通用性强，自动化程度高的方法，它的主要出发点是把软件用一个数学模型来刻划，对软件进行层次式功能分解，也对相应的模型函数进行分解，产生其子函数，这样下去产生层次式的软件结构。这种方法的特点也是对数据抽象重视不足[6]。这三种方法有一个共同的特点就是将功能抽象和数据抽象分开处理，从而导致软件结构和问题结构的不一致性，给以后的维护和理解带来困难。

从以上比较中可以看出，这些方法所产生的软件结构和人们的思维方式及问题结构都有一定的差距，用这些方法开发的软件在可理解性和可维护性上也有一定的问题。面向对象设计方法的出现，使得这种困难有所缓解，它是以抽象数据类型为中心的方法，它将数据抽象和功能抽象统一考虑，从而使软件模型和问题空间构成自然对应，在方法上合乎人们的思维逻辑，其目的就在于缩短问题空间中的给定问题和求解空间中的软件之间的距离，也就是说尽量使问题空间中问题的描述和解空间中软件的描述一致起来，从而提高软件的可理解性和易维护性。

在软件开发过程中，软件分析设计方法是主要因素，但仅仅依靠方法还不能设计出好的软件，因为软件开发是一个复杂的活动，必须要有一定的方便合理的描述工具和一些辅助工具来支持，这样才可以产生出高质量的软件，本文将介绍一个辅助于软件结构设计的工具——OOSSD-T。

一、软件结构设计过程

软件结构设计是以对象关系模型为基础的，它的整个设计过程是一个“对象关系模型建立→类层次结构建立→验证”的反复循环过程。首先，分析给定问题域，识别和确认存在的对象及相互之间的消息传递关系，建立给定问题域的初始对象关系模型，其次，将识别的对象进行分类，给出各类对象的描述，确立给定问题域的初始类层次结构，最后，对所建模型进行验证。以上三个子过程构成一次循环，如果新建模型还不是问题域的完整模型，则开始下一次循环，其相应的图示和具体步骤如下：

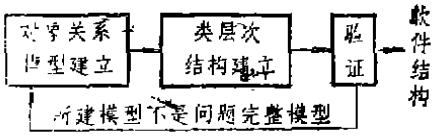


图1

第一阶段：对象关系模型的建立

本阶段的主要任务是分析给定问题，确立给定问题域的对象及相互间的消息传递关系，建立给定问题域的对象关系模型。

第一步：对象的确立

本步的任务是识别给定问题中存在的实体，从实体中确定对象及对象的性质。

第二步：建立对象关系模型

分析已识别的对象中各对象之间的关系，然后由关系确定对象之间的消息传递，传递什么样的消息，是接收消息，还是发送消息，最后形成对象关系网络，这种关系网络就是对象关系模型。

第二阶段：类层次结构的建立

本阶段的主要任务是将对象进行分类，给出各类对象的描述，建立给定问题的类层次结构。

第一步，类的确定

基于面向对象的软件设计主要是类的设计，而不是对象的设计，对象是在程序执行时动态产生的。因此，应将对象关系网络中的对象进行分类，其分类原则是：具有相同外部特性和内部处理能力的对象归为一类。在分类过程中可能出现两种情况：一种情况是由某些对象产生新类，另一种情况是某些对象归为已分好的类，如果是新类，则分析抽象该类的对象性质，给出该类的描述。

第二步，确定类的层次结构

所谓类的层次结构是类以继承关系建立起来的层次结构。它的确定根据不同情况可以综合采用下面三种方式：

1． 自底向上的方式

由对象直接抽象出来的类称为第一级类，对所有的第一级类进行分析，将具有公共性质的类的公共性质划入一个更大的类（super class），产生第二级类，即这些公共性质由第二级类来描述，被抽取公共性质的类作为子类（subclass），这些子类通过继承共享公共性质，如图2所示。然后对第二级类同样进行上述分析过程，产生更高级类，直到某一高级类有定义为止。

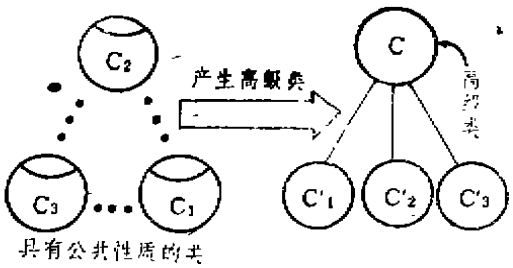


图2 Ci′通过对Ci抽取公共性质而得到

2． 自顶向下的方式

这种方式是从某一级类出发，将某个类的特殊性质划为一个子类，产生低级类，即特殊性质由低级类来描述，低级类通过继承共享它的高级类的性质，根据需要还可对低级类同样做上述分析工作，产生更低级的类，直到满足要求为止。其图式如下：

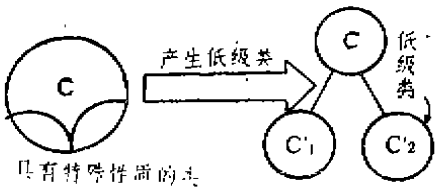


图3 类C_i'通过对C抽取特殊性质而得到

3． 分析已有的继承关系

分析在第一步所确定的类中是否有的类之间已具有一定的继承关系。

第三阶段：验证

验证所确定的模型是不是给定问题的完整模型。若不是则进行下一次循环。

本部分更详细内容请参阅[1][2][4][5]。

二、辅助工具（OOSSD-T）的实现

OOSSD-T是一个基于面向对象的软件结构设计辅助工具，它不仅支持用户设计软件结构的工作，而且还支持相应文档的设计和修改查阅（view）工作，文档由工具自动或交互式产生。它的用户界面是在一个高分辨彩色显示终端和多窗口支持下工作的。整个工具是在IBM PC/AT上用Microsoft C 4.0实现的。

2.1 辅助工具的功能和结构

OOSSD-T是一个多功能工具，它主要由以下几部分组成：当前工作状态选择、对象识别、结构编辑、软件结构的生成、文档管理、求助、多窗口及菜单管理、工具图形库、编辑功能函数库、类（class）文档库。

当前工作状态选择用于恢复和建立用户进行软件结构设计的工作状态；对象识别用于支持消息传送图（message passing diagram-MPD）的编辑、图形文档及实体对象关系模型的建立，同时这部分工作还受工具图形库，编辑功能函数库的直接支持，其中工具图形库中是一些具有特定意义的基本图形；结构编辑支持各种具有一定结构的正文文档的编辑工作，其中这些具有一定结构的正文文档都有其相应的结构模板，同时它还给用户提供了一系列方便的交互式编辑命令，如对图形、正文文档的建立、修改、删除、插入、查询等操作，在编辑过程中，可要强制地产生合乎一定规范的文档，并对这些文档进行一致性和语言方面的检查。

软件结构生成部分主要支持软件结构的产生和类文档的产生。用户对对象识别部分产生的结果进行分析抽象，用OOPSL语言进行描述，然后工具接收这种描述将产生软件结构图及相应的类文档；文档管理将对软件结构设计过程中所产生的图形和文本文档进行统一管理，提供对这些文档的查询、修改、打印、查看、权限等操作；求助部分能帮助用户熟练地使用辅助工具进行工作；多窗口及菜单管理部分主要是为增强用户界面的友好性而设计的，屏幕设计采用了多级选单驱动，菜单的弹出，窗口分区及覆盖，结构引导等技术；类文档库是为了减轻用户的工作量，以方便用户直接引用而设置的。

总之，OOSSD-T是一个多功能的工具，上述几个部分按功能可以归并为两个部分：分析设计和软件结构产生。用户在使用这个工具时，一般要有一个需求分析说明书或者对给定问题要做一定量需求分析工作，然后，用户可以按照需求分析结果，利用辅助工具来辅助以后的设计工作。OOSSD-T的系统结构如下图所示：

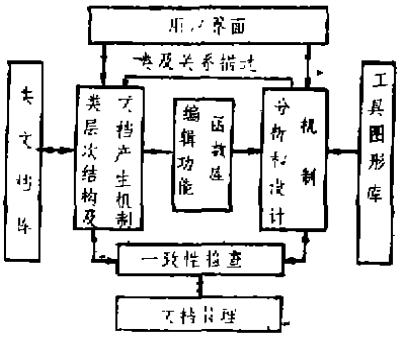


图4 OOSSD-T的系统结构

2.2 实体对象关系模型的建立

这部分工作是以用户为主体从需求分析说明书中识别实体对象，并在图形编辑和实体对象字典、消息字典的支持下完成的。图形编辑部分提供三个菜单：工具图形菜单，图形基本操作菜单，状态设置菜单，其中工具图形菜单和基本编辑操作菜单分别由工具图形库和编辑功能函数库来支持。

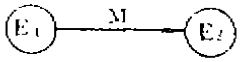
实体对象关系模型是一种实体对象关系网络，是由实体对象关系网络图，实体对象字典，消息字典来描述的。其中实体对象字典用来描述实体对象，消息字典用来描述实体对象之间的关系。

首先定义几个概念：

消息传送图：它是实体对象通过消息的发送和接收而联系起来的网络结构图，简称为"MPD"。

实体对象：它是给定问题域中实体的抽象体，包括对象或对象的类型。用带标记的圆符来表示。

消息用带标记的箭头表示，如



表示实体对象E_1向实体对象E_2发送消息。

MPD的分析及编辑过程是从给定问题域中识别实体对象及它们之间关系的过程。在此状态下，菜单提供表示实体对象的圆符和表示消息的箭头，用户可以对图形符号起名字或在图上写正文注释等，此外，用户可以对这些图形符号进行各种编辑操作，如，修改、删除、平移、拷贝、放大、缩小等，MPD的编辑支持层次的MPD结构，即对任何一个实体对象，可以使用分解操作把该实体对象分解为一层新的MPD，这层新的MPD由该实体对象的子实体对象所组成，该实体对象成为新图的父图，而新图则成为该实体对象的细节描述。

除此之外，还提供了与这些层次结构相应的操作，如，分层显示，分层编辑，分层存取等。

以上这种层次分析过程所产生的文档都将按其产生的顺序存放在文档树（文档的组织结构）相应的子树上，此子树可以体现原来的层次分析过程。

实体对象字典和消息字典是实体对象识别及层次分析技术的工具性字典，并且在整个编辑过程中自动建立，用户可对字典的某些项进行结构式编辑。

概括地说，OOSSD-T的对象识别机制工作结果如下图所示：

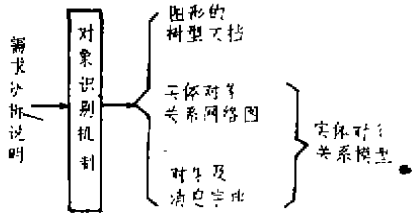


图5 实体对象关系模型的建立过程

2.3 类层次结构的产生

这部分的任务是由用户对实体对象关系模型中的实体对象做抽象化处理，给出类的描述，确定各类的继承关系，最后用OOPSL描述抽象化处理结果，由机器自动产生类层次结构图及相应的类文档。

OOPSL程序的BNF公式如下：

〈程序〉::=CLASS DESCRIPTION；

〈描述体〉

END

〈描述体〉::＝〈语句描述对〉{〈语句描述对〉}

〈语句描述对〉::＝〈CLASS关系描述语句〉

〈CLASS描述块〉

〈CLASS关系描述语句〉::＝〈CLASS关系定位项〉:

〈实体分类段〉

CLASS关系定位项〉::＝〈CLASS名，无符号整数〉

〈实体分类段〉::＝〈'obj:实体名表'〉

〈实体名表〉::＝〈实体名〉{，〈实体名〉}|空

〈CLASS描述块〉::＝

〈讨论说明〉

〈类名〉

〈父类名〉

〈类型定义〉

〈引用类名〉

〈实例变量说明〉

〈外部处理方法〉

〈内部处理方法〉

END

〈外部处理方法〉::＝

〈方法名〉

〈输入描述〉

〈输出描述〉

〈前置条件〉

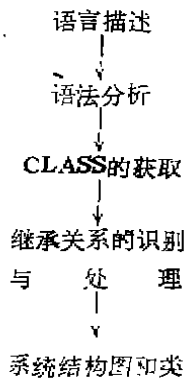
〈方法描述〉

〈功能描述〉

〈内部处理方法〉::＝同上

〈CLASS关系定位项〉中的无符号整数用来描述类之间的继承关系。

系统结构生成部分除了产生系统结构图之外，还产生相应的类文档树，其工作流程如下：



三、文档管理

至前面几个阶段所产生的文档都要在本部分得到统一管理。由于MPD图的编辑支持层次结构分析技术，生成的文档也是层次树型结构，类文档以继承关系形成层次树型结构，所以本部分就以文档树这种组织形式来对这些文档进行统一管理。它的管理方式是施行公开的方式，也就是说让用户在树上直接进行查阅、删除、权限等操作，这种公开方式是这样实现的：系统将文档树实体投影到屏幕上，其投影形式如图6，在屏幕上可以显示文档树投影的全貌，用户可以利用光标在文档树投影的节点上随便移动，当光标停在某一节点上时，就可以对这个节点进行操作，系统再将用户的操作从投影节点影射到实体文档树相应的节点上，从而达到文档管理操作的目的。文档树投影节点和实体文档树节点一一对应。将文档树的整个全貌呈现在用户面前是通过文档树窗口的上下，左右滚动来实现的。

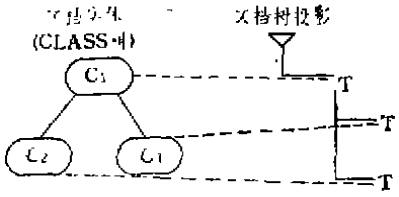


图6 文档实体的投影形式

T表示正文文档，G表示图形文档

文档管理屏幕还提供一个类文档库窗口和操作命令窗口，其中类文档库窗口方便用户查询自己所需要的CLASS文档描述，对库中的CLASS描述可以和文档树上的文档进行同样的操作。

本文前几部分系统地讨论了基于面向对象的软件结构设计方法和过程，还介绍了一个辅助于软件结构设计的工具及其实现特点。

面向对象设计方法是一个很有前途的方法，这种方法可以缩短问题空间和解空间之间的距离，因此，使用这种方法能够提高软件的生产率，增强软件的可理解性和易维护性，它的继承机制还支持代码一级的软件重用和设计重用。

整个工具已经实现完毕，并能正常运行，它是整个面向对象软件设计环境的一部分。从功能来说工具还有些不足，如，应该提高从对象关系模型到类的抽象自动化，这个工作还有待以后的改进。

本文曾得到吉林大学金淳兆老师的悉心指导，在此作者表示衷心感谢。

版本管理系统的设计方法

徐向阳

（华东师范大学计算机系）

摘　要

Version Management (VM) provides mechanisms for storing, accessing and inquirying each version, reducing space costs and recording modification information. An integrated VM design includes hierarchical designs of Configuration, module and text, VM tools should be rich in understanding of version semantics and relations to each other. A guideline for hierarchical design of VM is presented in this paper.

一、版本管理工具

计算机软件系统在其整个生命周期内需要经常修改、校正、扩充和改进，使系统产生多个版本，统称为系统簇。随着系统簇的不断膨胀，必须有效地管理和维护系统簇。软件配置管理SCM (Software Configuration Management) 是用来管理和维护系统簇的软件，其功能为：对源代码修改的控制和跟踪，配置生成，出错的跟踪与报表，以及产品的版本标志和管理。版本管理VM (version Management) 是支持SCM的一个工具，它应提供机制存贮、更新和查询软件的各个版本，减小软件版本所占存储空间，控制版本访问权限，以及记录版本形成时状态，如版本修改的原因、内容和修改者等。

先介绍两个概念：顺序版本和平行版本。

·顺序版本：因错误校正等原因，对版本进行一系列局部修改，按时间先后顺序生成的一连串文本。

·平行版本：由于软件模块的不同实现方式或功能扩展而形成一个以上的版本。

1. 源代码控制系统

最早的VM工具是在UNIX下运行的源代码控制系统SCCS（Source Code Control System, 1975），它是一个文本控制系统。为减小存贮开销，它只存放软件的初始版本和版本变更，为便于管理版本，限定版本访问权限和对版本附加标志和信息，并由程序带入目标文件；还可查询用户加入的注释信息和形成软件的修改报表。

2. 修改控制系统

AT & T公司于1982年发行了在UNIX下运行的修改控制系统 RCS (Revision Control System)。用户可指定文件名称和给出版本编号，以管

参考文献

1. Software Engineering Environment, Proceedings of the International Workshop on Software Engineering Environment. China Academic Publishers.
2. Adele Goldberg and David Robson. Smalltalk-80, The Language and its Implementation. Addison-Wesley, Reading, Massachusetts, 1983.
3. M. A. Jackson. "System Development", Englewood Cliffs, NJ. Prentice-Hall, 1983.
4. B. Stroustrup. "What is Object-Oriented Programming", IEEE Software, May 1988.
5. G. Booch. "Object-Oriented Development", IEEE Trans, Soft, Eng, Vol. SE-12, No. 2.
6. JAMES MARTIN. "SYSTEM DESIGN PROVABLY CORRECT CONSTRUCTS".