JBOO 软件开发过程*)

The JBOO Software Development Process

麻志毅

(北京大学计算机科学与技术系　北京100871)

Abstract The paper expatiates the JBOO software development process, explains briefly the software development tool JBOO 2.0 which supports the process entirely, and describes various models that JBOO 2.0 supports and the process uses.

Keywords Object-oriented, Software development process, Development tool

1 前言

JBOO 软件开发过程是青鸟工程中的一项重要成果,遵循青鸟面向对象软件开发规范。该规范包括了面向对象的软件开发概念与表示法、软件开发过程指导和软件开发文档编制三部分[1]。该规范也是青鸟面向对象软件开发工具JBOO2.0所支持的规范之一。

象其它的软件开发过程一样,JBOO 软件开发过程是为建造高质量的软件而制订的一系列活动,每个活动要完成若干任务。JBOO 软件开发过程中的重要因素有开发人员、项目、过程和工具。JBOO 软件开发过程以项目为中心,指导相关的人员协作进行软件开发。在 JBOO 软件开发过程中,项目是在生命期内面向机器、开发人员、管理人员和需方的所有制品的集合,项目的成果是一个最终的发布产品。过程是将用户需求转换成产品所需要的活动集的完整定义,过程还提供活动指南以及对产生的文档的要求。可以用支持 JBOO 软件开发过程的面向对象软件开发工具 JBOO2.0进行系统开发。

本文首先简介全面支持本过程的软件开发工具JBOO2.0,然后描述 JBOO2.0支持的、本过程要用到的各种模型,最后详细地阐述了 JBOO 软件开发过程。

2 面向对象的软件开发工具 JBOO 2.0

青鸟软件开发工具 JBOO2.0是一个可视化、详述、构造和文档化的集成化支持环境,支持青鸟面向对象规范。在该工具的支持下,对软件系统可进行可视化的建模。JBOO2.0支持从系统需求、系统分析、系统设计到代码生成的各个阶段,生成语法正确、语义完备并且一致的模型,以及相应的代码。这些工作由一系列的工具完成,请参见图1。图的上半部中的工具用于软件开发的三个过程,左下部中的工具用于支持复用,右下部中的工具用于支持软件开发。

对于一个面向对象的软件系统,可能要由多个模型进行描述。对每个模型可能要从不同的侧面、不同的抽象层次进行描述,这意味着每个模型可能由多张图组成。JBOO 软件开发过程涉及到用况模型、静态模型、行为模型和部署模型。

用况模型:描述系统功能和使用各功能的用户以及它们间的关系。

静态模型:描述系统中建模元素间的静态关系,强调系统的静态逻辑结构,即系统的组织。

行为模型:描述系统中建模元素间的动态关系,强调系统的动态方面。

部署模型:由系统的处理节点及分布在其上的系统软件组成,强调系统的静态分布。

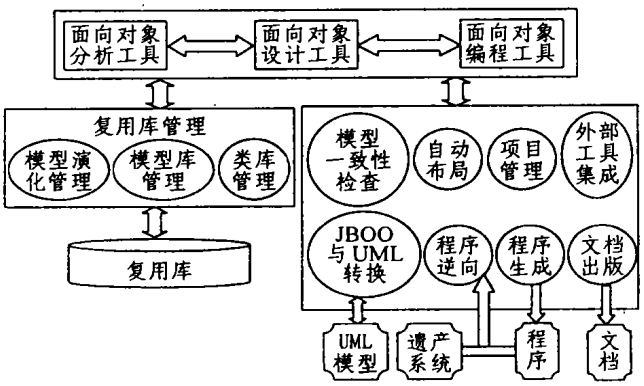

图1　青鸟面向对象软件工具集 JB002.0

模型的静态方面强调系统的结构性,它指明在整个应用的生命周期中不变的特征。模型的动态方面强调系统的行为性,注重于控制、时序和对事件的处理。图2描述了各模型的组成。

	用况模型	静态模型	行为模型	部署模型
静态方面	用况图	类图		部署图
动态方面			交互图、注重于消息的类图	

图2　模型的组成

随着软件项目规模的增大,单靠人来保证模型的各部分间的一致,图示与文档的一致,功能与结构的一致,几乎是不可能的。大型软件项目的各模型间的协作和约束关系就更为复杂。此外,模型是对现实世界中的事物的抽象,在某种程度上反映了客观现实,有主观性和相对性。因而,在建模过程中,JBOO2.0不但要提供动态的语法制导和语法检测功能,对语法的正确性进行检查,对建模中的众多信息,也要进行一致性检查。JBOO2.0与逆向工具相结合,还保证了模型和代码间的一致性。

*)本课题得到国家自然科学基金资助(项目编号60073015)。麻志毅　副教授,主要研究方向为软件工程与软件工程环境、面向对象方法。

3 软件开发过程

过程定义了谁在什么时间做什么以及如何达到特定的目标。一个软件开发过程当然要为合乎质量要求的软件开发提供有效的指导，为真实世界建立精确和正确的模型，但它也要简明、易于理解和使用，这是制订 JBOO 软件开发过程的一个原则。

JBOO 软件开发过程分为软件需求描述、面向对象分析(OOA)、面向对象设计(OOD)、面向对象编程(OOP)和面向对象测试(OOT)阶段。可把线性顺序模型、增长模型和演化模型等用于本过程，但本过程注重于迭代增量开发。

下面分别阐述各个阶段所涉及到的开发人员、他们要进行的活动和完成的任务以及产生的制品。

3.1 软件需求描述

进入此阶段应该完成了业务模型或领域模型，准备好了系统规格说明书，进行了系统结构设计，即完成了标识硬件配置项、软件配置项和人工操作项，并把系统需求分配到各项中。以这些工作为基础，结合问题域形成软件规格说明书[2](描述各软件项的功能与能力、外部接口等)。图3描述了完成软件需求描述所要进行的输入与输出，以及参与的人员。

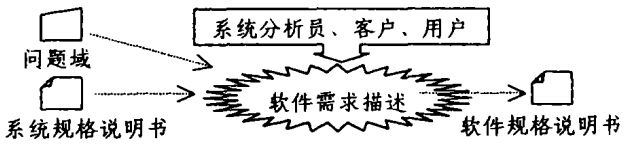

图3 软件需求描述中的活动者及输入/输出

进行软件需求描述后，应保证所有的软件需求要反映在软件规格说明书中。它是以后各项工作的输入。

3.2 面向对象的分析

结合问题域，针对软件规格说明书中的各软件项，进行软件结构设计。把一个软件项分为若干个软件子系统，对每个子系统进行面向对象的分析工作。图4描述了面向对象分析中的活动模型。

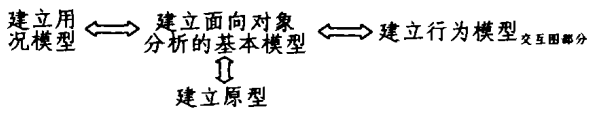

图4 面向对象分析中的活动模型

3.2.1 *建立用况模型* 首先针对每个软件子系统建立用况模型。该项活动包含如下的任务：(1)确定系统边界；(2)定义参与者；(3)定义用况；(4)建立用况图。

根据系统的情况建立多个用况图，形成用况模型。图5描述了建立用况模型所要进行的输入与输出，以及参与的人员。

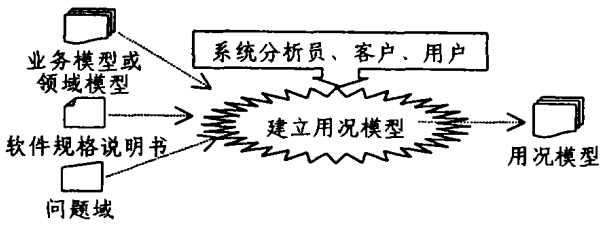

图5 建立用况模型的活动者及输入/输出

3.2.2 *建立面向对象分析的基本模型* 面向对象分析的基本模型包括静态模型和部分行为模型，它们都是由若干个类图组成。用这些类图从不同的角度对系统建模。静态模型注重于类间的静态关系，而行为模型注重于类间的消息连接。

在确定系统的用况后，可使用 CRC(class responsibility collaborator)技术，辅助发现类以及类之间的协作关系。用 CRC 模拟用况，也检查了用况模型的正确性。

该项活动针对于问题域建模，要进行如下的工作：(1)发现对象，建立类；(2)定义类的属性与服务；(3)定义类间的结构与连接；(4)划分主题。

对于大型系统，可在简单地认识到子系统中的一些主要对象后即可划分主题，据此进行任务分工，协作开发。

图6描述了建立 OOA 基本模型所进行的输入与输出，以及参与的人员。

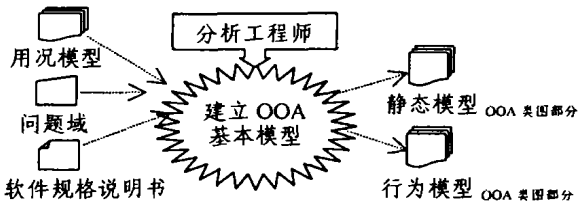

图6 建立 OOA 基本模型的活动者及输入/输出

3.2.3 *建立原型* 在确定了基本模型中的主要对象后，就可建立原型。如果需要，随着分析的深入，可进一步地建立原型。

3.2.4 *建立行为模型OOA交互图部分* 行为模型OOA交互图部分由一组交互图组成。每个交互图详细且直观地描述了一组对象之间以及一组对象和参与者之间的关系。图7描述了建立行为模型所要进行的输入与输出，以及参与的人员。

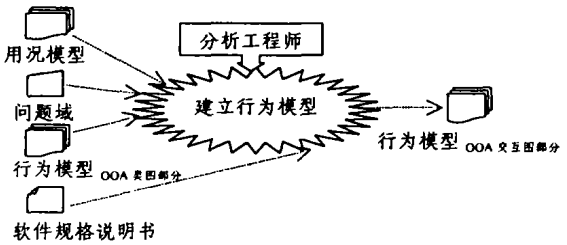

图7 建立行为模型的活动者及输入/输出

在定义并复审了所有的类、类的属性和操作、类之间的结构关系、行为模型和可复用的类后，即可进入面向对象设计阶段。

3.3 面向对象的设计

在面向对象的设计阶段，对分析阶段的结果进行设计，但注重考虑与实现有关的因素。此外还要进行人机交互、任务管理、数据管理和系统部署方面的设计。

此阶段要做的具体工作同3.2.2和3.2.4小节，只是侧重点不同。对各软件子系统中的类及其间关系进行详细设计，要细化到能被编码的程度。软件子系统的需求应分派到各个类。对软件子系统的外部接口和内部的部署件之间的接口也要进行详细设计。

3.3.1 *问题域设计* OOA 阶段建立的基本模型和行为模型OOA部分到 OOD 阶段不需要转换，但要决定类间关系的实现方式、考虑编程语言对设计的影响以及对例外进行处理，可

能还要对基本模型中的多继承进行调整。例如，用什么样的数据结构实现属性、实现各方法的算法和设计消息等都是此阶段要进行的工作。

图8描述了进行问题域设计时所要进行的输入与输出，以及参与的人员。

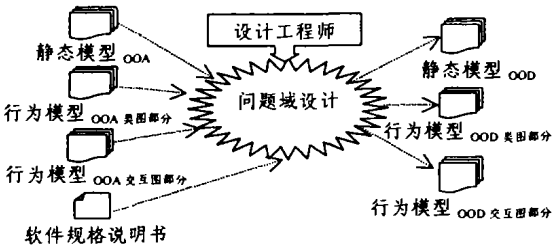

图8　进行问题域设计的活动者及输入/输出

3.3.2　数据管理设计　为了存储问题域中永久对象，本部分包括对存储和检索对象的基本结构的设计以及对数据库的设计。

对存储和检索对象的基本结构进行设计包括对永久对象类的存储设计、对实例连接的存储设计、对整体部分结构的存储设计、对一般-特殊结构的存储设计和对相应类的服务设计等。关于数据库的设计方面的资料有很多，请参看有关文献。图9描述了进行数据管理设计时所要进行的输入与输出，以及参与的人员。

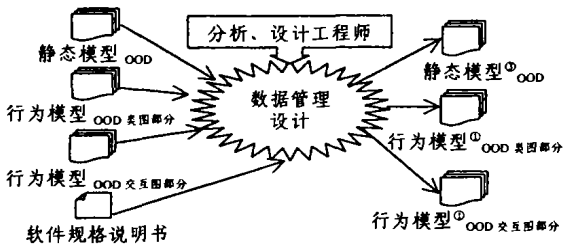

图9　与数据管理设计相关的活动者及输入/输出

静态模型②OOD、行为模型②OOD类图部分和行为模型②OOD交互图部分都在已有的基础上，增加了数据管理部分以及数据库本身的设计部分。

3.3.3　人机界面交互　本部分要按用户的要求和选定的 GUI 设计系统进行人和软件系统的交互界面的设计，包括初始命令层、报告及报表和窗口等设计。图10描述了进行人机界面设计时所进行的输入与输出，以及参与的人员。

静态模型②OOD、行为模型②OOD类图部分和行为模型②OOD交互图部分都是在静态模型OOD类图部分、行为模型OOD类图部分和行为模型OOD交互图部分的基础上，增加了人机界面交互设计部分。

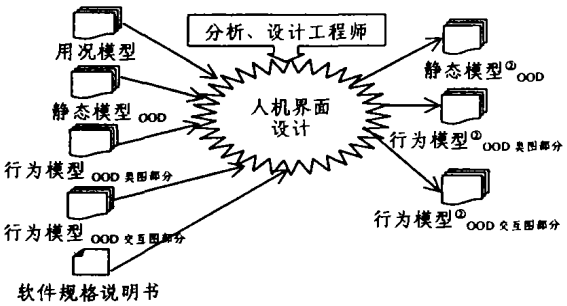

图10　与人机界面设计相关的活动者及输入/输出

3.3.4　控制接口设计　为本系统与其它系统间的接口以及本系统控制的物理设备的接口进行建模，但计算机的一般外设（如显示器和打印机等）在此通常不予与考虑。此阶段要识别由事件驱动的任务、识别由时钟驱动的任务以及识别优先任务和关键任务等。图11描述了进行控制接口设计时所要进行的输入与输出，以及参与的人员。

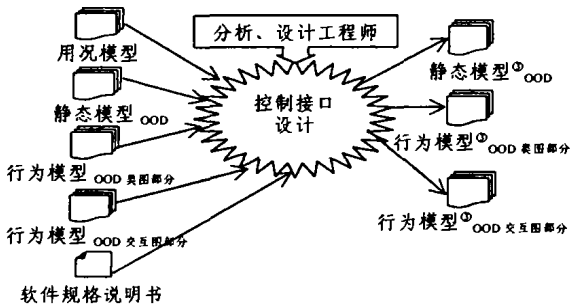

图11　与控制接口设计相关的活动者及输入/输出

静态模型③OOD、行为模型③OOD类图部分和行为模型③OOD交互图部分都是在静态模型OOD类图部分、行为模型OOD类图部分和行为模型OOD交互图部分的基础上，增加了控制接口设计部分。

3.3.5　系统部署设计　系统部署设计是对类的组织和对对象的分布所做的进一步考虑。对于各子系统，在进行了上述的四项工作后，对产生的类图要进行划分主题的工作。在此处要考虑这样的主题：其中的各元素协作完成相同的功能并驻留在相同的硬件产品中。结合对子系统的集成考虑，需要建立系统的部署模型。即要按照系统的分布情况，对模型元素进行组织，再把它映射到节点上。这要完成如下三项工作：

·定义节点。节点是拥有某些计算资源的物理对象（设备）。根据系统功能，决定设立哪些节点，并进行定义，其中包括它们的能力（处理能力、内存大小等）、连接的类型和通讯协议等。

·定义部署件。一般把一个主题作为一个部署件。对部署件进行描述，把实现部署件的类作为部署件规约的一部分。

·绘制部署图。指明各个部署件要放在哪个节点上。一个部署件可以分布在一个或多个节点上。

图12描述了进行系统部署设计时所要进行的输入与输出，以及参与的人员。

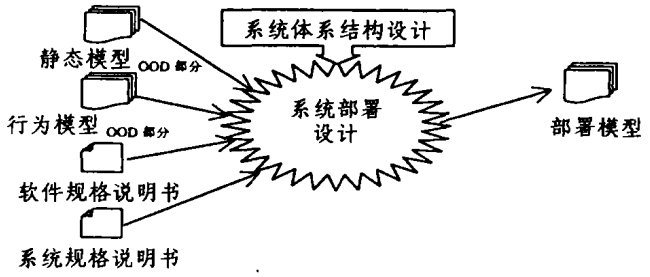

图12　与系统部署设计相关的活动者及输入/输出

静态模型OOD部分和行为模型OOD部分是上述3.3.1～3.3.5小节四项工作所产生的模型的统称。

3.4　系统实现

按照已有的设计，对各类进行编码和测试，识别出作为系统物理实现部分的附属部署件（如数据表、文件和文档等），并针对各源代码文件生成可执行文件，部署到相应的节点上，产

（下转第110页）

$$G(p,f^{(n)})=D(\tilde{p}\|p^{(n-1)})-D(\tilde{p}\|p^{(n)}) \quad (15)$$

因此选择的第 n 个特征为：

$$f^{(n)}=\underset{f\in F}{\operatorname{argmax}}\, G(p,f^{(n)}) \quad (16)$$

模型选取特征是从特征空间这里看作是候补特征集合中选取的，因此我们首先定义一个特征空间，也可以看作是选取特征的模式。特征空间的定义和选取对于问题的处理也是一个关键。目前经过加工的语料有词，词性标注或语法信息都包含在语料中，我们根据不同的任务定义不同的特征空间。例如，在词性标注时我们定义特征空间，这里选取当前词 w_i 的前后两词及前两词的词性，即：$\{w_i, w_{i+1}, w_{i+2}, w_{i-1}, w_{i-2}, t_{i-1}, t_{i-2}\}$，然后特征空间具体化，如果将所有特征都考虑在内，则数量极其庞大，这样我们就可以根据上面介绍的 FI 算法选择一个最佳的特征集合加入到最大熵模型中。最后对于选取的每个特征都赋予一个权值，也就是我们模型的参数。这时就可以根据模型进行预测将来的行为。由于篇幅有限，本文不再详细介绍。

结束语 应用最大熵模型的统计方法可以把模型与语言学知识分为两个模块进行处理，在机器翻译[8]、词性标注[9]、语法分析[10]、部分分析[11]等各项语言处理任务中都得到了有效的应用，模型可以被多次利用，对不同的任务只是选择的特征不同。最大熵框架的这种通用性和重用性允许实验者使用其他任务中相同的参数估计程序。参数估计的代码实质上与特定任务无关，一个实现就可以满足所有的其他任务模型。更重要的是，最大熵模型在每个任务中都表现得相当好，尽管所有任务在本质和复杂性上是不同的。各种实验结果表明研究者可以使用和重用最大熵框架到非常广泛的任务中去，并具有很高的准确性。此外，最大熵模型本身对于任务的处理还有以下优点：

1)控制细微结果。使用最大熵可以准确为变量间的细微依赖关系建模，在建立最大熵模型时，可以跨距离地选取特征。这样选取的特征用传统的预测模型技术是不可能的。

2)不做未经验证的假设。最大熵承认已有的事实，对所选特征没有独立性假设。而传统的预测模型例如决策树、逻辑回归和神经网络对信息都会做一些错误的假设。

3)该模型简单、易于理解。本文中的特征集不需要深层的语言学知识，只问上下文中的基本问题。比其他方法较少依赖语言学知识、预处理、或语义数据库，因此更容易指定和移植特征。尽管特征明显的简化，它们仍可以有效地近似复杂的语言学关系。

本文对最大熵模型及其算法进行了详细的介绍，对于汉语处理中的各项任务，需具体问题具体分析，分别选取不同的特征集合结合到该模型中，而无须过多涉及到模型和算法问题，因此该模型框架对于汉语信息处理具有实际的意义。

参 考 文 献

1 Skut, Wojciech, Brants T. A Maximum Entropy Partial Parser for Unrestricted Text. In: 6th Workshop on Very Large Corpora, Montreal, Canada, Aug. 1998

2 周强. 规则和统计相结合的汉语词类标注方法. 中文信息学报, 1995, 9(3)

3 周强. 一个汉语短语自动界定模型. 软件学报, 增刊, 1996, 7: 315～322

4 Ratnaparkhi A. Maximum Entropy Models for Natural Language Ambiguity Resolution: [Ph. D.]. Dissertation, University of Pennsylvania, 1998

5 Darroch J N, Ratcliff D. Generalized Iterative Scaling for Log-Linear models. Annals of Mathematical Statistics, 1972, 43(5): 1470～1480

6 Pietra S D, Pietra V D, Lafferty J. Inducing features of random fields, IEEE Transactions on Pattern Analysis and Machine Intelligence, 1997, 19(4): 380～393

7 Berger A. The Improved Iterative Scaling Algorithm: A Gentle Introduction. http://www. cs. cmu. edu/afs/cs/user/aberger/www/ps/scaling. ps, 1997

8 Berger A, Pietra S D, Pietra V D. A maximum entropy approach to natural language processing. Computational Linguistics, (22-1), March 1996

9 Ratnaparkhi A. A maximum entropy model for part-of-speech tagging. In: Proc. of the Conf. on Empirical Methods in Natural Language Processing, 1996

10 Skut, Wojciech, Brants T. A Maximum Entropy Partial Parser for Unrestricted Text. In: 6th Workshop on Very Large Corpora, Montreal, Canada, Aug. 1998

11 Erik F, Sang T K, Buchholz S. Introduction to the CoNLL-2000 Shared Task: Chunking. In: Proc. of CoNLL-2000 and LLL-2000, Lisbon, Portugal, 2000

(上接第135页)

生部署模型$_{实现}$。部署模型$_{实现}$由一些从实现角度建立的部署图组成。图13描述了进行系统实现设计时所进行的输入与输出，以及参与的人员。

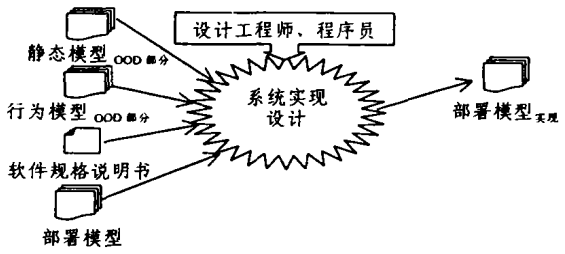

图13 与系统实现设计相关的活动者及输入/输出

3.5 系统测试

JBOO 软件开发过程中的系统测试可采用各种面向对象的测试方法。

结束语 通过对国际上相关规范的研究与应用[3～5]，我们深深地体会到一套软件开发过程规范应该易学易用，并注重实效。

JBOO 软件开发过程是我们在多年的软件工程研究和开发实践中总结出来的成果。在具体应用中可根据情况进行裁减。我们还开发了一套 JBOO 软件开发工具，它支持本开发过程。

参 考 文 献

1 青鸟工程资料. 青鸟面向对象软件开发规范. 北京大学计算机科学技术系, 2000

2 International Standard. Information technology- Software Life Cycle Processes. ISO/IEC 12207

3 Jacobson I, Booch G, Rumbaugh J. The Unified Software Development Process. Addison-Wesley Object Technology Series, 1999

4 UML documentation version 1. 3. http://www. rational/uml/index. jsp

5 Sellers B H, Firemith D G. Comparing OPEN and UML: the two third-generation OO development approaches. Information and Technology, 1999, 41: 139～159