A Integrity Specification by OO Visual Modeling of UML— A Case Based Study in Computing Education *

YI Yang, MA Fei-teng, RONG Fu-li, LI Xiao-xing
(Information Science School & Technology, Sun Yat-sen University, Guangzhou 510275, China)

Abstract: Objected-oriented based software analysis and design with visual modeling (OOVM) has become a mature technology, meanwhile, the Unified Modeling Language (UML) is a diagrammatic notation widely taught in universities as a way to represent software requirements specifications and design descriptions. We address a whole process of software system analysis and design by objected oriented methodologies and identify several problems in teaching software system analysis and design with visual modeling by UML. A meta-model of Medical Assistant Diagnosis Support System (MADSS) in UML is offered, and complete integration of OOVM concepts is also discussed.

Key words: software engineering computing education; objected-oriented visual modeling; UML
CLC number: TP31 Document code: A Article ID: 0529-6579 (2007) S2-0074-05

1 Introduction

This paper addresses those calls by offering an illustration about every process in the software system analysis & design by using UML, based on a practical case which is a Medical Assistant Diagnosis Support System (MADSS). The offers complete the integration of all OOVM concepts.

2 Problem Statement and Requirement Overview

The process requirement engineering is starting at a Problem Statement (PS) documentary provided by the customer. The PS of MADSS is described as follows:

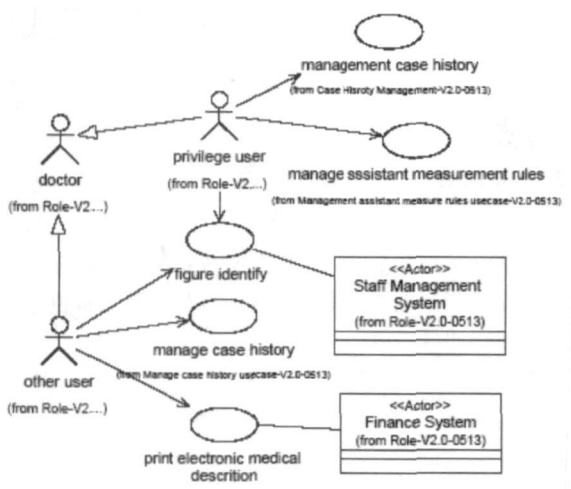

Fig. 1 Use-case Model of MADSS

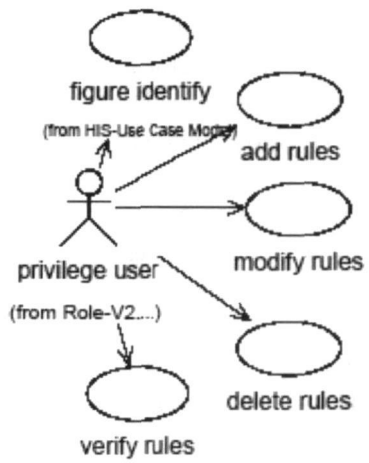

Fig. 2 Fundamental Event in MAMRU

MADSS is a medical information management system to be built to replace the legacy system. The main objective of MADSS is to gather more medical case data which is confirmed to the Federal Electronic Medical Case Criterion and improve the diagnosis efficiency with the help of computer assistant measure. The doctors of the hospital can input, modify or inquire the medical cases online via the intranet, meanwhile, make use of the Medical Assistant Measurement Rules. The privilege users (doctors) are authorized to define the medical case history template. The other users (doctor) are accredited to select the template and record the diagnosis record with it.

The legacy system only deals with registering pa-

*收稿日期: 2007-10-28
作者简介: 衣杨 (1967年生), 女, 副教授; E-mail: issyy@mail.sysu.edu.cn

tient and the basic human resource information of the doctors. Customer (the hospital) recommends that the legacy system should be reserved to keep on managing the Human Resources, meanwhile the new implemented MADSS and the legacy system should share a common database.

The relevant artifacts of Requirement Overview (RO) are as follows:

1) Use-case Model

The use-case model of MADDS is shown in Fig. 1, and the Fig. 2 is the model of Management Assistant Measurement Rules Use-case (MAMRU).

2) Use-case Specification

Since the space is limited, only the MAMRU. The main function of MAMRU is: accept the assistant measure rules as candidate-rules provided by the privilege user, then verify the candidate-rules by comparing the candidate-rules with the details in medical cases which are randomly selected from whole case history; if the coincidental rate is bigger than 95%, the candidate-rules will be employed as a assistant measurement rules and will be memorized.

a) Brief Introduction: The privilege users manage the assistant measurement rules; the other users take the use of the rules in their diagnosis. b) Event Flow: i) The basic event is shown in Fig. 2, and the event flow can be found in Fig. 8. ii) The adjuvant event flow includes of Reject the candidate rules. c) Pre-condition: The privilege users login in the system. d) Post-condition: New assistant measurement rules are created.

3　Analysis Engineering

3.1　Architecture Analysis

The Brower/Server architecture is required by the customer, so a multi-layer (4 layers) pattern is presented by Fig. 3.

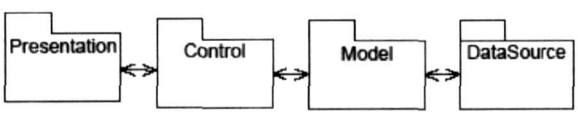

Fig. 3　Architecture Framework of MADSS

3.2　Key Abstraction

The Key Abstraction is a concept and must be handled, normally uncovered in requirements. Its objective is to identify the class. Its sources are domain knowledge, requirements glossary, domain model, or the business model (if exists). The Key Abstraction is illustrated by VOPC Diagram (View of the Participant Class). The VOPC of the MAMRU is provided in Fig. 4.

3.3　Use-case Analysis

The process of use-case analysis is to identify the classes which perform a use-case flow of events, distribute the use-case behavior to those classes, and identify responsibilities of the classes.

The artifact of use-case analysis for MAMRU is presented in Fig. 5. Analysis classes are the classes demonstrated by the extended UML. The boundary classes are the classes who intermediate between the interface and something outside the system, so one boundary class per actor/use-case pair. The entity classes are those in the Key Abstraction, which are used to store and manage information in the system. The control classes are use-case behavior coordinators which are use-case dependent and environment independent.

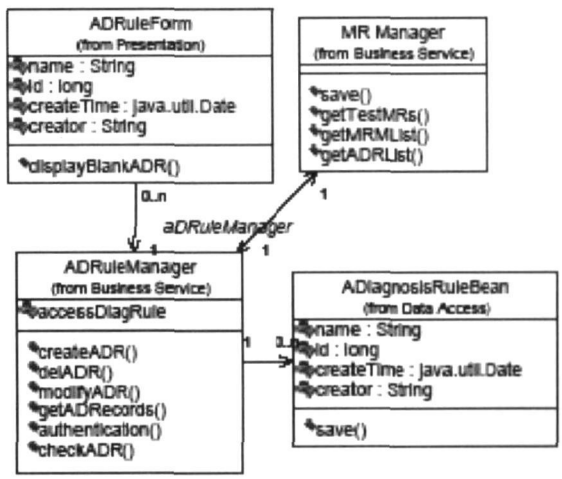

Fig. 4　Key Abstraction of the MAMRU

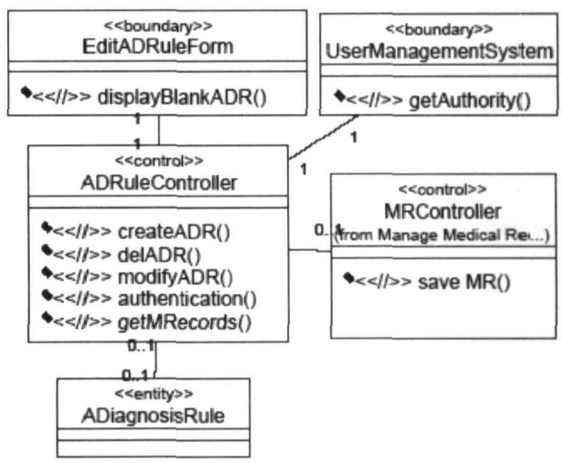

Fig. 5　Use-case Analysis of the MAMRU (refined class)

4　Design Engineering

4.1　Use-case Design

The objectives of use-case design are to refine the use-case realization, identify subsystems and their in-

terface, identify reuse opportunities and update the organization of the design model, such as architecture or design pattern. The Architecture Framework of MADSS illustrated by Fig 3 is improved to the model shown in Fig 6. The result of use-case design for MADSS is developed in Fig 7 and Fig 8. More detailed refined use-case model is presented in Fig 9 as an example.

Interface classes represent the model elements that realize the interfaces, but no message should be drawn from the interface. However, Proxy classes can represent a specific subsystem in which messages can be drawn from the proxy.

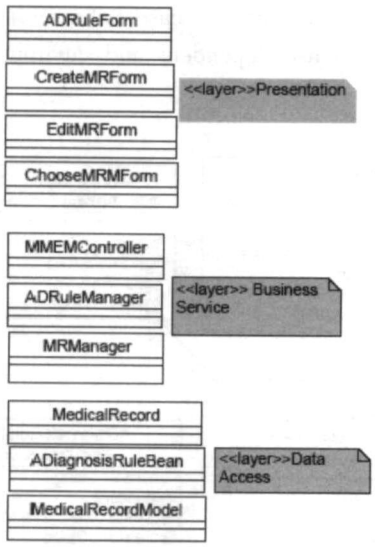

Fig. 6 Updated Architecture Framework of MADSS

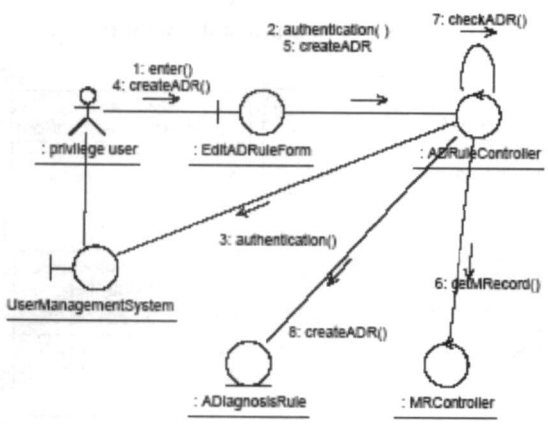

Fig. 7 Collaboration Diagram of MAMRU

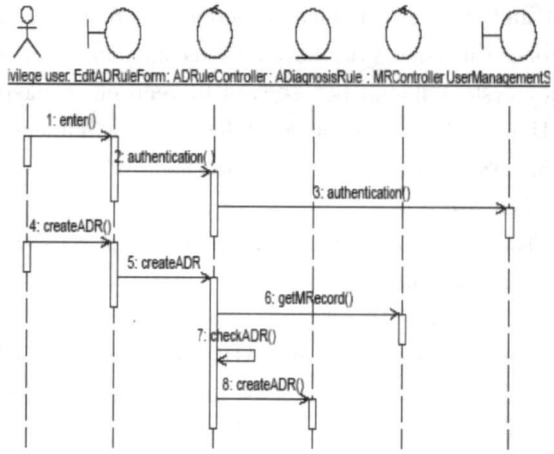

Fig. 8 Sequence Diagram of MAMRU

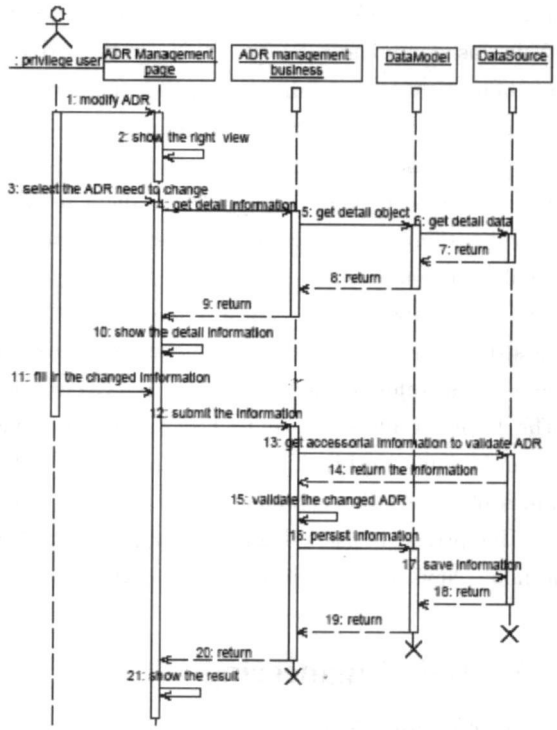

Fig. 9 Sequence of Add Assistant Measurement Rule Case

4.2 Subsystem Design

A subsystem is a "cross between" a package and a class. Its responsibilities are defined by interface operations to model interface realizations. And interface operations may be realized by internal class operations or internal subsystem operations. One or more interaction diagrams demonstrate per interface. The subsystem and legacy interface realization diagram is shown by Fig 10.

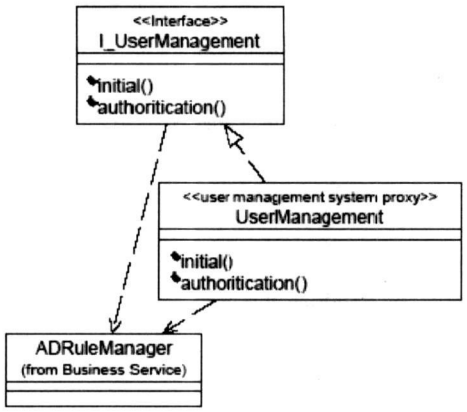

Fig. 10　Subsystem & Legacy System Interface

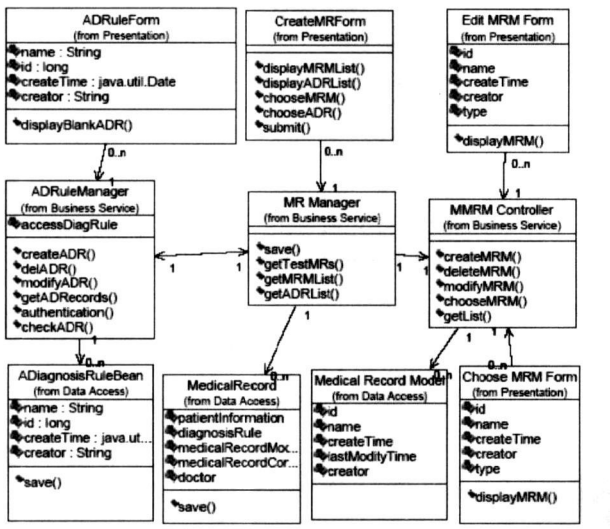

Fig. 11　Designed Class Diagram of MAMRU (subsystem)

4.3　Class Design

In class design process, following aspects should be considered. Class stereotype, which includes boundary class, entity class and control class, are described by extended UML; applicable design patterns, such as modularity, proxy, etc; architectural mechanisms, such as persistence, distribution, etc. The class diagram of MADSS and the whole system of (MADSS) are presented in Fig. 11 and Fig. 12 respectively.

5　Conclusion

A case based study on software system analysis & design based on objected oriented methodology with visual modeling language UML is discussed, and the full scale specification with documentary and diagrams are presented. It should be noted that the whole process is an iterative development which is made of several iterations on analysis and design.

We conclude some valuable strategies in class analysis & design. For the boundary classes designing, the user interface (UI) boundary classes should consider the problem of what user interface development tools will be used and how much of the interface can be created by the development tool should be considered; on the other hand, the external system interface boundary classes can be usually model as subsystem. For the designing on entity classes, entity objects are often passive and persistent, such as performance requirements, the persistent classes. For the control classes designing, the existence necessary of the control class should be thought over; meanwhile, the characters of complexity, change probability, distribution and performance and transaction management should be considered.

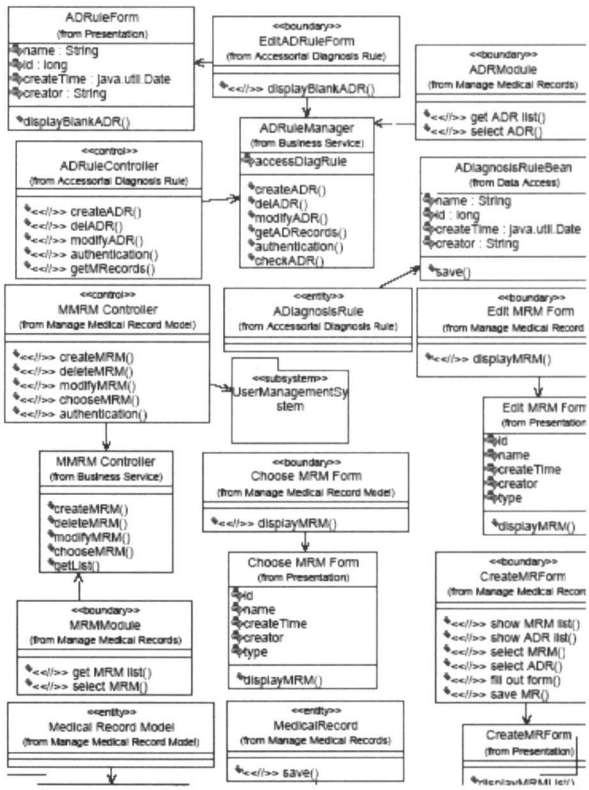

Fig. 12　Designed Class Diagram for MADSS

Many simple classes mean that each class must encapsulates less of the overall system intelligence, more reusable and easier to implement. Few complex classes mean that each class must encapsulates a large portion of the overall system intelligence, less likely to be reusable and more difficult to implement. So, a class should have a single well-focused purpose and do one thing and do it well.

6　Acknowledgement

This paper is supported by the National Natural Science Foundation of China (60573159), Natural Science Foundation of GuangDong (05200302) and the Education Project of SYSU.

References:

[1] APVRILLE L, COURTIAT J P, LOHR C, et al. Saqui-Sannes: TURTLE: A Real-Time UML Profile Supported by a Formal Validation Toolkit [Transactions on Software Engineering][J]. IEEE Computer Society, 2004, 473—487.

[2] LAVAGNO L, MARTIN G, SELIC B. UML for Real: Design of Embedded Real-Time Systems[M]. Kluwer, 2003.

[3] MELLOR S, BALCER M. Executable UML: A Foundation for Model-Driven Architecture[M]. Addison-Wesley, 2003.

[4] OMG. Unified Modeling Languages Specification, version 1.4, 2001.

[5] OMG. Unified Modeling Languages Specification, version 1.5, 2003.

[6] QUATRANI T. Visual Modeling with Rational Rose and UML[M]. Addison-Wesley, 1998.

结合案例分析基于 UML的面向对象可视化建模教学

衣 杨，容福丽，马飞腾，李晓星

（中山大学信息科学与技术学院，广东 广州 510275）

摘 要：面向对象软件可视化分析与设计建模（OOVM）已成为一个较为成熟的技术，而 UML是目前高校软件工程教学中广泛使用的一个图形建模工具。基于一个案例，描述了面向对象软件分析与设计的全过程，指出了基于 UML建模方法所需输出的必要制品，讨论了 OOVM集成的完整技术。

关键词：计算机软件工程教育；面向对象可视化建模；UML

中图分类号：TP31