R

PERPUSTAKAAN UNIVERSITI MALAYA

ACI - 0443
INVC. 17/1/02

DEVELOPMENT OF A REUSABLE OBJECT-ORIENTED APPLICATION FRAMEWORK FOR LIBRARY SYSTEMS DOMAIN

ROY CHEW TECK CHOYE

A thesis submitted
in fulfilment of the requirement for the
Master of Software Engineering

September 2000

Perpustakaan Universiti Malaya
A510575210

Acknowledgement

This research project has been produced with the enormous help that I received from people directly or indirectly involved with it. Firstly, the most influential person on this research project undertaken is my project supervisor, Dr. Lee Sai Peck, who in very own individualistic way carried out her supervision with admiration, respect and commendation from me. Without her, this research project and dissertation would not be possible. A big thank you to her for her patience, expertise advice and quick feedback.

I would also like to take this opportunity to thank very much the many people involved on the Internet who helped me to solve the many software engineering problems that I faced during my research project. A lot of discussions on newsgroups, forums and emails have proven themselves to be significant for quick assistance. Long live Internet! I would also like to thank Sun for Java, POET for POET database and Editplus for its programmer's editor. I hope to repay you someday.

Roy Chew
September, 2000

Table of Contents

List of Figures........ vi

Abstract........ viii

1 Introduction........ 1

- 1.1 Motivations........ 3
- 1.2 Problem Statement........ 4
- 1.3 Objectives of Research........ 5
- 1.4 Outcome of Research........ 6
- 1.5 Thesis Organisation........ 6

2 Literature Review........ 8

- 2.1 Application Frameworks........ 8
  - 2.1.1 Background and Definitions........ 8
  - 2.1.2 Collaborating Classes........ 11
  - 2.1.3 Class Library Versus Framework........ 12
  - 2.1.4 Constituents of Framework........ 14
    - 2.1.4.1 Software Components........ 14
    - 2.1.4.2 Robust Components........ 20
  - 2.1.5 Other Types of Frameworks........ 21
    - 2.1.5.1 Architectural Framework........ 21
    - 2.1.5.2 Technical Framework........ 22
- 2.2 Identification of Domain's Functions........ 22
- 2.2.1 Common Features (Commonalities)........ 24
- 2.2.2 Variable Features (Variabilities)........ 24
- 2.3 Design Mechanisms........ 26
- 2.4 Subsystems via Unified Modeling Language (UML) Package. 30
- 2.5 Survey of Library Systems Domain........ 33
- 2.6 Research Proposal : Development of an OO Application Framework for Library Systems Domain........ 34

2.6.1 Research Methodology 35
2.6.2 Research Constraints 40
2.7 Summary 41

3 OO Application Framework on Library Domain: An Architectural View 42
3.1 Layered Architecture 42
3.1.1 Application Systems Layer 43
3.1.2 Business-Specific Layer 44
3.1.3 Middleware Layer 44
3.1.4 System Software Layer 45
3.2 A Family of Library Applications 45
3.2.1 Library Application Systems Layer 45
3.2.2 Library Systems Domain Layer 46
3.2.3 Middleware Layer for Library Systems Domain 47
3.2.4 System Software Layer for Library Systems Domain... 47
3.3 Summary 48

4 Analysis and Design of OO Application Framework on Library Systems Domain 49
4.1 Domain Analysis on Library Systems Domain Model 49
4.1.1 Library Systems Requirements with Use Case Modelling 52
4.1.2 Library Systems Architecture with Analysis Modelling.. 55
4.1.2.1 Superordinate System Model 56
4.1.2.2 Superordinate Design Model 59
4.2 System Design on Library Systems Domain Model 62
4.2.1 High-level Scenario of System Design 62
4.2.2 Static Structure with Class Diagrams 64
4.2.3 Dynamic Structure with Interaction Diagrams and State Diagrams 66

4.3 Summary 73

5 Framework Implementation 74

5.1 Implementation Classes 76

5.2 Configuration and Initialisation of Business Rules 77

5.3 Storage Management 79

5.4 Application Framework Customisations 81

5.5 Application Framework Evaluation 85

5.6 Summary 94

6 Conclusion 96

6.1 Strength of Research 98

6.2 Limitations of Research 100

6.3 Future Work 101

6.4 Concluding Remarks 101

References 103

Appendix

Appendix I : Installation Guide 105

Appendix II : User Guide 107

List of Figures

Figure 2.1: Software component layers... 17

Figure 2.2: Types of Partitioned Business Components (Andersen) 18

Figure 2.3: Extension of variation point.. 25

Figure 2.4: Interacting components.. 30

Figure 2.5: Package diagram... 32

Figure 2.6: Software Reuse Life Cycle Model.................................. 38

Figure 3.1: A typical layered software system................................. 43

Figure 3.2: Family of applications on top of a layered system architecture.. 46

Figure 4.1: Use Case model of library system requirements............ 53

Figure 4.2: Use Case model of library requirement variation point... 54

Figure 4.3: Superordinate Use Case in the superordinate system... 56

Figure 4.4 : Traceability of a Use Case to an analysis model component in a Loan subsystem................................ 57

Figure 4.5: Superordinate analysis types mapped into several subsystems.. 58

Figure 4.6: Library Systems Superordinate Design Model................ 60

Figure 4.7: An application system import from a facade.................. 61

Figure 4.8: Superordinate Design Model input into a high-level sequence diagram.. 63

Figure 4.9: Class Diagram for Library Systems Domain.................. 65

Figure 4.10: Sequence Diagram for a Loan process....................... 67

Figure 4.11: State Diagram for an Item.. 68

Figure 4.12 : State diagram for a Title reservation........................... 69

Figure 4.13 : Object Interactivity Diagram (Sequence Diagram) for borrowing an Item... 70

Figure 4.14 : Object Interactivity Diagram (Sequence Diagram) for returning an Item... 71

Figure 4.15 : Object Interactivity Diagram (Sequence Diagram) for a Title Reservation.. 72

Figure 5.1 : Layered system architecture.. 74

Figure 5.2: Initialisation file for configuration of library entity status and business rules.. 78

Figure 5.3: Step 1 of Wizard tool: Choosing the framework's application database........ 82

Figure 5.4: Step 2 of Default Option in Step 1 Wizard tool.............. 83

Figure 5.5: Step 2 of Customise Option in Step 1 Wizard tool......... 84

Figure 5.6: Insert a Title dialog.. 86

Figure 5.7: Insert a Borrower dialog.. 87

Figure 5.8: Insert a Loan dialog... 87

Figure 5.9: Information Retrieval of a Borrower............................... 88

Figure 5.10: Information Retrieval of a Title..................................... 89

Figure 5.11: Information on a Title object with Reservation status... 90

Figure 5.12: Onhold status for Reservation..................................... 91

Figure 5.13: Fine on Borrower for overdue Loan............................. 92

Figure 5.14(a): Delete a Borrower Dialog.. 93

Figure 5.14(b): Information Feedback of Deletion on Borrower....... 93

Abstract

The main goal of developing an application framework on an application domain or a business domain is to facilitate development of application systems within the same domain by reusing components through customisation and not from scratch. These components would have to be analysed and designed with reusable features so that they can be reused to develop different applications in the related domain. In the case of an object-oriented application framework in the technical view, it is made up of a set of interrelated abstract (generic) and concrete classes that underlie the components of a domain model and form the domain architecture that can be reused and customised by application developers to develop a specific application system on the same domain.

An application framework basically produces reusable code or software components that reduce software development time. The aim of producing a more reliable software by reusing code is the primary aim of object-oriented programming. Instead of writing an application from scratch, application development using a framework is a better way because the reusable software components have been tested. In this research project, we will study the ways to define the abstraction details of the library systems domain to be included into a set of generic framework classes and the means to develop wizards to assist application developers in using the framework. We will also prove the use of the framework by building a sample application that is fully object-oriented, which uses an object-oriented database for data persistency. The common and variable features of library systems domain will be identified in the requirements analysis phase and designed in the software components that exist in the layered software architecture of the framework. In the layered architectural view, interrelated components are grouped together as a logical namespace identified as "package". Each package is identified with a purpose to develop a subsystem or system for a domain. By sorting out a framework into packages, application developers can quickly identify the components for reuse.

Software reuse can be rewarding in terms of quicker application system development and reliability. The convenient reuse of software packages can help to achieve that aim.