ABSTRACT

Design patterns represent an important evolutionary step in software abstraction and reuse. Design patterns are solutions to general software development problems. One interesting area in this field is the automatic detection of design pattern in existing code. This could be of significant help to the process of reverse engineering software systems.

Thus, the main objective of this research is to create a tool that detects design patterns. This involves studying the unique characteristics (rules) of each of the design patterns in order to detect them and built these rules into the tool. The tool also produces visualization of the detected patterns in terms of UML class diagram.

The scope of this study is to detect design patterns that fall under the Structural category. These patterns are Adapter, Bridge, Composite, Decorator, Façade, Flyweight, and Proxy. Besides, it is focusing on detecting design patterns in Java code.

This study resulted in a tool called Design Pattern Detection and Visualization (DPDV). DPDV accepts Java files as inputs from users. It then extracts the static and structural relationships between classes and objects that existed in the Java code. The tool uses the Reflection technique in order to do that. The tool then arranges the extracted information in a data structure defined as a class. After that, the tool examines the extracted information and compared them with the rules of each of the design patterns to determine the possible design pattern that existed in the Java code. The pattern can be detected since each structural pattern has a unique structure that differentiates it from other structural patterns. The tool achieved a detection/accuracy rate that exceeded 85% that indicates that it is successful.
The analysis, design, implementation and testing of DPDV, its strengths and weaknesses are also discussed in this study. As a conclusion, this study has achieved its objectives. This will definitely benefit similar future study and developers of such a system.
ACKNOWLEDGMENTS

Many people provided encouragement and inspiration that led to the creation of this research. I particularly thank my research supervisor for her invaluable guidance, support, and encouragement throughout this research.

My special thanks and respect to my beloved family, who have been the ultimate source of my motivation to work hard; therefore, I proudly dedicate this work to them.

Finally, I thank all my friends who have been supporting and motivating me throughout this work.

April 2008
Wael M K Fares
# TABLE OF CONTENTS

## CHAPTER 1: INTRODUCTION

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1 Introduction</td>
<td>1</td>
</tr>
<tr>
<td>1.2 Problem Statement</td>
<td>1</td>
</tr>
<tr>
<td>1.3 Design Patterns</td>
<td>3</td>
</tr>
<tr>
<td>1.4 Research Objectives</td>
<td>3</td>
</tr>
<tr>
<td>1.5 Scope of study</td>
<td>4</td>
</tr>
<tr>
<td>1.6 Research Significance</td>
<td>4</td>
</tr>
<tr>
<td>1.7 DPDV Platform, Hardware and Software Requirements</td>
<td>4</td>
</tr>
<tr>
<td>1.7.1 DPDV Hardware Requirements</td>
<td>5</td>
</tr>
<tr>
<td>1.7.2 DPDV Software Requirements</td>
<td>5</td>
</tr>
<tr>
<td>1.8 Research Methodology</td>
<td>5</td>
</tr>
<tr>
<td>1.9 Overview of the report</td>
<td>6</td>
</tr>
</tbody>
</table>

## CHAPTER 2: LITERATURE REVIEW

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1 Introduction</td>
<td>9</td>
</tr>
<tr>
<td>2.2 History of design patterns</td>
<td>9</td>
</tr>
<tr>
<td>2.3 Structural Design patterns</td>
<td>10</td>
</tr>
<tr>
<td>2.3.1 Adapter</td>
<td>11</td>
</tr>
<tr>
<td>2.3.2 Bridge</td>
<td>11</td>
</tr>
<tr>
<td>2.3.3 Composite</td>
<td>12</td>
</tr>
<tr>
<td>2.3.4 Decorator</td>
<td>13</td>
</tr>
<tr>
<td>2.3.5 Façade</td>
<td>14</td>
</tr>
<tr>
<td>2.3.6 Flyweight</td>
<td>15</td>
</tr>
<tr>
<td>2.3.7 Proxy</td>
<td>15</td>
</tr>
<tr>
<td>2.4 Object-Oriented features for patterns realization and detection</td>
<td>15</td>
</tr>
<tr>
<td>2.5 Examples of Design Patterns</td>
<td>17</td>
</tr>
<tr>
<td>2.5.1 The composite pattern</td>
<td>17</td>
</tr>
<tr>
<td>2.5.2 The decorator pattern</td>
<td>21</td>
</tr>
<tr>
<td>2.6 Existing Detection systems and techniques</td>
<td>23</td>
</tr>
</tbody>
</table>
2.6.1 Existing Detection systems

2.6.1.1 Pat system

2.6.1.2 JBOORET (Jade Bird Object-Oriented Reverse Engineering Tool)

2.6.1.3 DPVK (Design Pattern Verification toolkit)

2.6.1.4 SPQR: Flexible Automated Design Pattern Extraction from Source Code

2.6.2 Existing Detection techniques

2.6.2.1 Bit-vector algorithm

2.6.2.2 Negative search criteria

2.6.2.3 Multi-stage reduction strategy using software metrics

2.6.2.4 Static and Dynamic Analyses combined

2.6.3 Comparison among the existing detection systems and techniques

2.7 Visualization techniques

2.7.1 Programs visualization

2.7.2 Visualizing design patterns using UML

2.7.3 Visualizing design patterns using virtual reality and hypertext

2.8 Reflection technique for design patterns detection

2.8.1 Definition of Reflection

2.8.2 Mechanism of Reflection

2.8.3 Reflection Example

2.9 Summary

CHAPTER 3: METHODOLOGY

3.1 Introduction

3.2 Input gathering and analysis

3.3 Design and Development

3.3.1 Development model for DPDV

3.3.2 Design

3.4 Evaluation

3.5 Summary
CHAPTER 4: DESIGN PATTERNS DETECTION AND VISUALIZATION ANALYSIS

4.1 Introduction 51
4.2 Rules of the structural design patterns 51
  4.2.1 Adapter design pattern rules 51
    4.2.1.1 Object Adapter design pattern rules 51
    4.2.1.2 Class Adapter design pattern rules 54
  4.2.2 Bridge design pattern rules 55
  4.2.3 Composite pattern rules 57
  4.2.4 Decorator design pattern rules 59
  4.2.5 Façade design pattern rules 61
  4.2.6 Flyweight design pattern rules 62
  4.2.7 Proxy design pattern rules 63
4.3 DPDV Detection and Visualization Techniques 65
  4.3.1 DPDV Detection Technique 65
  4.3.2 DPDV Visualization technique 66
4.4 System Analysis 66
  4.4.1 Functional requirements 66
    4.4.1.1 DPDV use case diagram 66
    4.4.1.2 Use case specification 67
  4.4.2 Non-Functional requirements 74
4.5 Summary 75

CHAPTER 5: SYSTEM DESIGN

5.1 Introduction 76
5.2 DPDV architecture design 76
5.3 DPDV Class Diagram 77
5.4 DPDV Sequence Diagram 79
5.5 Data structure design 80
5.6 Algorithm design 84
5.7 User interface design for DPDV 86
5.8 Summary 93
CHAPTER 6: SYSTEM IMPLEMENTATION

6.1 Introduction 94
6.2 Usage of design patterns in the implementation of DPDV 94
6.3 Use of Object-Oriented Programming in the implementation of DPDV 96
   6.3.1 Use of member access controls 96
   6.3.2 Use of composition 96
6.4 DPDV code explanations 97
   6.4.1 ‘Dialog1’ class 97
   6.4.2 ‘FilesLoader’ class 97
   6.4.3 ‘Detector’ class 99
   6.4.4 ‘Structure’ class 101
   6.4.5 ‘ReadMethods’ class 104
   6.4.6 ‘Visualizer’ class 105
   6.4.7 ‘ClassDetails’ class 105
6.5 Summary 107

CHAPTER 7: SYSTEM TESTING

7.1 Introduction 108
7.2 Testing stages 108
   7.2.1 Preparation of test data 108
   7.2.2 Unit testing 111
   7.2.3 Integration testing 111
   7.2.4 System testing 112
   7.2.5 Acceptance testing 115
7.3 Evaluation of DPDV 116
7.4 Summary 117

CHAPTER 8: DISCUSSION OF RESULTS AND CONCLUSION

8.1 Introduction 118
8.2 Discussion of results 118
8.3 Weaknesses and Strengths of DPDV 119
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>8.3.1 Strengths</td>
<td>119</td>
</tr>
<tr>
<td>8.3.2 Weaknesses</td>
<td>120</td>
</tr>
<tr>
<td>8.4 Future Work</td>
<td>120</td>
</tr>
<tr>
<td>8.5 Conclusions</td>
<td>121</td>
</tr>
<tr>
<td>REFERENCES</td>
<td>122</td>
</tr>
</tbody>
</table>
LIST OF FIGURES

Figure 2.1: Composite pattern diagram (Bansiya, 1998) 17

Figure 2.2: Example of composite design pattern: relationship between container and component 18

Figure 2.3: Use of the Composite pattern to nest AWT containers (Geary, 2001) 19

Figure 2.4: Use of the composite pattern (Geary, 2001) 20

Figure 2.5: Nested AWT containers (Geary, 2001) 21

Figure 2.6: Use the Decorator pattern (Geary, 2001) 23

Figure 2.7: Architecture of the Pat System (Kramer and Prechelt, 1996) 25

Figure 2.8: JBOORET Architecture (Mei et al., 2001) 26

Figure 2.9: The structure of DPVK (Wang and Tzerpos, 2005) 27

Figure 2.10: SPQR Outline (Smith and Stotts, 2003) 29

Figure 2.11: User interface screen-shot of Antoniol’s system (Antoniol, 2006) 31

Figure 2.12: Static and Dynamic Analyses of Patterns (Heuzeroth et al., 2003) 33

Figure 2.13: Four axes of program visualization (Roman et al., 1992) 38

Figure 2.14: Overall System Architecture (Dong et al., 2005) 40

Figure 2.15: Input XML File for Class Diagram (Dong et al., 2005) 40

Figure 2.16: Connection Pool Class Diagram with the Pattern Information (Dong et al., 2005) 41

Figure 2.17: The three spatial dimensions (Callaghan et al., 1998) 43

Figure 2.18: Reflection example 45

Figure 3.1: Evolutionary Prototyping model (Sommerville, 2007) 49
Figure 4.1: Object Adapter design pattern structure diagram

(Gamma et al., 1995) 53

Figure 4.2: Object Adapter design pattern code (using interface implementation) 53

Figure 4.3: Object Adapter design pattern code (using class inheritance) 53

Figure 4.4: Class Adapter design pattern structure diagram (Gamma et al., 1995) 54

Figure 4.5: Class Adapter design pattern code 54

Figure 4.6: Bridge design pattern structure diagram (Gamma et al., 1995) 56

Figure 4.7: Bridge design pattern code (using interface implementation) 56

Figure 4.8: Bridge design pattern code (using class inheritance) 57

Figure 4.9: Composite design pattern structure diagram (Gamma et al., 1995) 58

Figure 4.10: Composite design pattern code (using interface implementation) 58

Figure 4.11: Composite design pattern code (using class inheritance) 58

Figure 4.12: Decorator design pattern structure diagram (Gamma et al., 1995) 60

Figure 4.13: Decorator design pattern code (using interface implementation) 60

Figure 4.14: Decorator design pattern code (using class inheritance) 61

Figure 4.15: Façade design pattern structure diagram (Gamma et al., 1995) 61

Figure 4.16: Façade design pattern code 62
Figure 4.17: Flyweight design pattern structure diagram (Gamma et al., 1995)  63

Figure 4.18: Flyweight design pattern code  63

Figure 4.19: Proxy design pattern structure diagram (Gamma et al., 1995)  64

Figure 4.20: Proxy design pattern code  64

Figure 4.21: DPDV use case diagram  67

Figure 4.22: The concrete quality scenario for the reliability (accuracy) quality of the DPDV system  74

Figure 5.1: DPDV Architecture design  77

Figure 5.2: DPDV class diagram  79

Figure 5.3: DPDV Sequence Diagram  80

Figure 5.4: DPDV main screen  87

Figure 5.5: DPDV Files Loader screen  88

Figure 5.6: Open files dialog box (standard design)  88

Figure 5.7: Loaded files  89

Figure 5.8: Design Patterns Detector screen  90

Figure 5.9: Visualization (class diagram) screen  91

Figure 5.10: Classes Details visualization (textual form) screen  92

Figure 6.1: Use of Façade design pattern in DPDV’s implementation  95

Figure 6.2: Use of member access control in ‘ReadMethods’ class  96

Figure 6.3: Use of composition relation in class ‘Detector’  97

Figure 6.4: ‘FilesLoader’ class (code to load files)  98

Figure 6.5: ‘FilesLoader’ class (code for opening output and input files)  99

Figure 6.6: ‘Detector’ class (code to extract classes using reflection)  100
Figure 6.7: ‘Detector’ class (code to extract interfaces methods using reflection) 101
Figure 6.8: ‘Detector’ class (code to extract classes’ methods using reflection) 101
Figure 6.9: Matching extracted information against Bridge pattern rules in
   ‘Structure’ class 103
Figure 6.10: ‘CheckMethod’ method in ‘ReadMethods’ class 104
Figure 6.11: ‘Visualizer’ class (code to draw bridge design pattern) 105
Figure 6.12: ‘Visualizer’ class (code to display class details when loading
   the class) 106
Figure 6.13: ‘Visualizer’ class (code to display classes’ details when mouse
   is clicked) 107
Figure 7.1: Overall accuracy/detection rate (%) for each DPDV version 115
**LIST OF TABLES**

Table 2.1: Comparison among the existing detection systems 34
Table 2.2: Comparison among the existing detection techniques 36
Table 4.1: Load java files Use case 68
Table 4.2: Detect patterns Use case 68
Table 4.3: Extract attributes and relationships Use case 69
Table 4.4: Arrange info in data structure 70
Table 4.5: Map between extracted info and pattern rules 71
Table 4.6: View patterns textually Use case 72
Table 4.7: View patterns Visually Use case 72
Table 4.8: View classes details Use case 73
Table 5.1: Fields and methods of ‘Structure’ class 81
Table 5.2: Pseudo code of DPDV system 84
Table 7.1: Set 1 Java code samples (Stelting and Maassen, 2002) 109
Table 7.2: Set 2 Java code samples 110
Table 7.3: Set 3 Java code samples (Truett, 1998) 110
Table 7.4: Testing Results of the DPDV Version 1 113
Table 7.5: Testing Results of the DPDV Version 2 113
Table 7.6: Testing Results of the DPDV Version 3 114
Table 7.7: Characteristics of DPDV system 116
### LIST OF ABBREVIATIONS

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AOL</td>
<td>Abstract Object Language</td>
</tr>
<tr>
<td>API</td>
<td>Application Program Interface</td>
</tr>
<tr>
<td>AST</td>
<td>Abstract Syntax Tree</td>
</tr>
<tr>
<td>AWT</td>
<td>Abstract Window Toolkit</td>
</tr>
<tr>
<td>BON</td>
<td>Business Object Notation</td>
</tr>
<tr>
<td>DPDV</td>
<td>Design Pattern Detection and Visualization</td>
</tr>
<tr>
<td>DPVK</td>
<td>Design Pattern Verification toolkit</td>
</tr>
<tr>
<td>GB</td>
<td>Gigabytes</td>
</tr>
<tr>
<td>GHz</td>
<td>Gigahertz</td>
</tr>
<tr>
<td>GUI</td>
<td>Graphical User Interface</td>
</tr>
<tr>
<td>HCI</td>
<td>Human-Computer Interaction</td>
</tr>
<tr>
<td>HTML</td>
<td>Hyper Text Markup Language</td>
</tr>
<tr>
<td>JBOORET</td>
<td>Jade Bird Object-Oriented Reverse Engineering Tool</td>
</tr>
<tr>
<td>OMT</td>
<td>Object Modeling Technique</td>
</tr>
<tr>
<td>OO</td>
<td>Object-Oriented</td>
</tr>
<tr>
<td>OOP</td>
<td>Object-oriented programming</td>
</tr>
<tr>
<td>OOPSLA</td>
<td>Object-Oriented Programming Systems, Languages, and Application</td>
</tr>
<tr>
<td>SPQR</td>
<td>System for Pattern Query and Recognition</td>
</tr>
<tr>
<td>UML</td>
<td>Unified Modeling Language</td>
</tr>
<tr>
<td>XML</td>
<td>eXtensible Markup Language</td>
</tr>
</tbody>
</table>