MEDiator CONNECTor FOR COMPosing LOOSELY COUPLED SOFTWARE COMPONENTS

Hamid Sanatnama

FSKTM 2009 6
MEDIATOR CONNECTOR FOR COMPOSING LOOSELY COUPLED SOFTWARE COMPONENTS

By

Hamid Sanatnama

Thesis Submitted to the School of Graduate Studies, University Putra Malaysia in Fulfilment of the Requirement for the Degree of Doctor of Philosophy

June 2009
DEDICATION

To my dearest wife and my lovely children for the endless support, encouragement and patience, and also those who made this study possible.
Abstract of thesis presented to the Senate of University Putra Malaysia in fulfillment of the requirement for the degree of Doctor of Philosophy

MEDIATOR CONNECTOR FOR COMPOSING LOOSELY COUPLED SOFTWARE COMPONENTS

By

HAMID SANATNAMA

June 2009

Chairman: Associate Professor Abdul Azim Abdul Ghani, PhD
Faculty: Computer Science and Information Technology

Component-Based Software Development (CBSD) is an approach that has many benefits, such as improving application developer productivity, reducing costs and complexity by reusing of existing codes. Programming within this approach is like assembly (i.e. composing software out of prefabricated components) rather than development, which reduces skill requirements, and allows expertise focuses on domain problems. The foundation of any CBSD methodology is its underlying component model, which defines what components are, how they can be constructed, and specifies the standards and conventions that are needed to enable composition of independently developed component. The current component models do not support composition in both design and deployment phase. They also focus on the specification and packaging of components but provide almost no support for the easy composition of components. Component in these models uses either direct or indirect message passing as connection schemes, which leads to tightly coupling (i.e. components mix computation with
control). It is concluded that this research has proposed an effective way for component composition which provides loosely coupling between composed components. For system maintenance and evolution, this decoupling should make it simpler to manage changes in the components, and also changes in the connector separately. This research has resulted in the proposed of mediator connector which is similar to a communication hub. It initiates method calls and manages the returns, and also provides total loosely coupling between components and also itself. Mediator connector is a framework and can be reused without any modification. The components composition using mediator connector belongs to the deployment phase. Our approach is based on interactions between components as a subset of behavior in a system. In order to minimize coupling between components and mediator connector, we have designed and developed an XML-based language, called Component Interaction Markup Language (CIML), where components as well as their interactions are described in a CIML document which are used by mediator connector. For evaluation of mediator connector in order to measure the loosely coupling it provides, four case studies have been tested. To measure coupling we applied Coupling Between Object (CBO) software metric. The result shows that mediator connector provides totally loosely coupling between software components composing in the deployment phase.
Abstrak tesis yang dikemukakan kepada Senat Universiti Putra Malaysia sebagai memenuhi keperluan untuk ijazah Doktor Falsafah

PENGHUBUNG PENENGAH UNTUK MENGKOMPOS KOMPONEN PERISIAN KEBERGANTUNGAN LONGGAR

Oleh

Hamid Sanatnama

June 2009

Pengerusi: Profesor Madya Abdul Azim Abd. Ghani, PhD
Fakulti: Sains Komputer dan Teknologi Maklumat

Pembangunan Perisian Berasaskan Komponen (PPBK) adalah satu pendekatan yang mempunyai banyak faedah, seperti mempertingkatkan produktiviti pembangun aplikasi, mengurangkan kos dan kompleksiti dengan menggunakan semula kod yang ada. Pengaturcaraan dalam kaedah ini adalah seperti pemasangan (iaitu mencantumkan perisian menggunakan komponen yang telah terhasil) tidak seperti pembangunan, yang mana mengurangkan keperluan terhadap kemahiran, dan membenarkan pakar untuk memfokus kepada permasalahan utama. Asas kepada mana-mana metodologi CBSD adalah model komponen asasnya, yang mana mentakrifkan apakah dia komponen, bagaimana untuk membinanya, dan menspesifikasi piawaian dan peraturan yang diperlukan untuk membolehkan pembentukan komposisi komponen yang dibangunkan berasingan. Model komponen semasa tidak menyokong komposisi dalam kedua-dua fasa reka bentuk dan penyerahan. Mereka memfokus kepada spesifikasi dan pembungkusan komponen tetapi hampir tidak menyokong untuk mempermutahkan komposisi bagi komponen. Komponen dalam model ini menggunakan penyerahan mesej sama ada
kelonggaran kebergantungan yang disediakan, empat kajian telah dilakukan. Untuk mengukur kebergantungan kami menggunakan metrik perisian ‘Coupling Between Objejct’ (CBO). Keputusan menunjukkan bahawa penengah penghubung menghasilkan kebergantungan yang sangat longgar antara komponen perisian yang dikompos semasa fasa penyrahan.
ACKNOWLEDGEMENTS

This dissertation would not have been possible without the kindness of several individuals. I would like to express my gratitude and deepest appreciation to my supervisor, Assoc. Prof. Dr. Abdul Azim Abdul Ghani for his kindness and significant mentoring all the time and for offering his guidance and insightful comments in the aspects of developing mediator connector using CIML. Assoc. Prof. Mohd. Hasan Selamat also deserve a great deal of thanks not only for his practical guidelines, but also for sponsoring my PhD study through the Graduate Research Assistant (GRA) scheme for his research, and Dr Rodziah Atan for her kindness, and helpful advices specially for starting my thesis writing.

I am also deeply thanking all the staffs working in Faculty of Computer Science and Information Technology in UPM, especially technical staff.

My special thanks go to my dearest wife, Mahnaz Radboo, my children Elnaz and Iman for their understanding and patience throughout the course of my research and the preparation of this dissertation. I would also like to express my uttermost appreciation to Assoc. Prof. Dr. Faramarz Sadeghi, and Assoc. Prof. Dr. Hamid Khosravi from Faculty of Mathematic and Computer Science in Shahid Bahonar University of Kerman, who made the opportunity for me to continue my study. And also a great deal of thanks to all others who have helped me in one way or another, specially my best friend Reza Meimandi Parizi.
I certify that a Thesis Examination Committee has met on 19 June 2009 to conduct the final examination of Hamid Sanatnama on his thesis entitled “Mediator Connector for Composing Loosely Coupled Software Components” in accordance with the Universities and University College Act 1971 and the Constitution of the Universiti Putra Malaysia [P.U.(A)] 15 March 1998. The Committee recommends that the student be awarded the Doctor of Philosophy.

Members of the Thesis Examination Committee were as follows:

**Name of Chairperson, PhD**
Associate Professor Dr. Azmi bin Jaafar
Faculty of Computer Science and Information Technology
Universiti Putra Malaysia
(Chairman)

**Name of Examiner 1, PhD**
Associate Professor Dr. Hamidah Ibrahim
Faculty of Computer Science and Information Technology
Universiti Putra Malaysia
(Internal Examiner)

**Name of Examiner 2, PhD**
Associate Professor Dr. Ali Mamat
Faculty of Computer Science and Information Technology
Universiti Putra Malaysia
(Internal Examiner)

**Name of External Examiner, PhD**
Professor Dr. Safa’ai bin Deris
Faculty of Computer Science and Information Systems
Universiti Teknologi Malaysia
Johor Malaysia
(External Examiner)

---

**BUJANG KIM HUAT, PhD**
Professor and Deputy Dean
School of Graduate Studies
Universiti Putra Malaysia

**Date:** 13 July 2009
This thesis submitted to the Senate of Universiti Putra Malaysia and has been accepted as fulfilment of the requirement for the degree of Doctor of Philosophy. The members of the Supervisory Committee are as follows:

**Abdul Azim Abdul Ghani, PhD**  
Associate Professor  
Faculty of Computer Science and Information Technology  
Universiti Putra Malaysia  
(Chairman)

**Mohd. Hasan Selamat**  
Associate Professor  
Faculty of Computer Science and Information Technology  
Universiti Putra Malaysia  
(Member)

**Rodzial Binti Atan**  
Associate Professor  
Faculty of Computer Science and Information Technology  
Universiti Putra Malaysia  
(Member)

__________________________

**AINI IDERIS, PhD**  
Professor/Dean  
School of Graduate Studies  
Universiti Putra Malaysia

Date:
DECLARATION

I hereby declare that the thesis is based on my original work except for quotations and citations which have been duly acknowledged. I also declare that it has not been previously or concurrently submitted for any other degree at UPM or other institutions.

HAMID SANATNAMA

Date: 30 June 2009
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABSTRACT</td>
<td>iii</td>
</tr>
<tr>
<td>ABSTRAK</td>
<td>v</td>
</tr>
<tr>
<td>ACKNOWLEDGEMENTS</td>
<td>viii</td>
</tr>
<tr>
<td>APPROVAL</td>
<td>ix</td>
</tr>
<tr>
<td>DECLARATION</td>
<td>xi</td>
</tr>
<tr>
<td>LIST OF TABLES</td>
<td>xv</td>
</tr>
<tr>
<td>LIST OF FIGURES</td>
<td>xvi</td>
</tr>
<tr>
<td>LIST OF ABBREVIATIONS</td>
<td>xxii</td>
</tr>
</tbody>
</table>

## CHAPTER

### 1 INTRODUCTION

1.1 Background

1.1.1 Composition

1.1.2 Composition at Design Phase

1.1.3 Composition at Deployment Phase

1.1.4 Composition Scheme

1.1.5 Exogenous Connectors

1.2 Problem Statement

1.3 Objectives

1.4 Scope of the Study

1.4.1 Evaluation and Verification

1.5 Contribution of This Study

1.6 Research Methodology

1.7 Outlines of this Dissertation

### 2 BACKGROUND IN COMPONENT-BASED SOFTWARE DEVELOPMENT AND COMPOSITION

2.1 Introduction

2.2 Component-Based Software Development

2.3 Composition

2.3.1 Scripting Languages

2.3.2 Composition Languages

2.3.3 Wrapping

2.3.4 The Software Bus

2.3.5 Connectors

2.4 Loosely and Tightly Coupling

2.4.1 Design Patterns

2.4.2 Creational Patterns

2.4.3 Structural Patterns

2.4.4 Behavioral Patterns

2.4.5 Mediator Design Pattern
3 RELATED WORK
  3.1 Introduction 32
  3.2 Component Models 33
    3.2.1 JavaBeans 34
    3.2.2 Enterprise Java Beans 36
    3.2.3 CORBA 40
    3.2.4 COM 48
    3.2.5 Koala 57
    3.2.6 SOFA 62
    3.2.7 Architecture Description Languages (ADLs) 66
    3.2.8 Exogenous Connectors 73
  3.3 Summary 74

4 RESEARCH METHODOLOGY 77
  4.1 Introduction 77
  4.2 Literature Study on Software Component Models 78
  4.3 Literature Study on Software Composition Techniques 79
  4.4 Design and Development of a Connector 80
  4.5 Evaluation of Mediator Connector 81
  4.6 Summary 82

5 COMPONENT INTERACTION MARKUP LANGUAGE (CIML) 83
  5.1 Introduction 83
  5.2 Requirements for CIML 84
  5.3 CIML Specification 86
  5.4 Extensible MarkUp Language 86
    5.4.1 Overview 86
    5.4.2 What is XML? 87
    5.4.3 XML editor 88
    5.4.4 XML Structure Definition 88
  5.5 EBNF grammar for CIML 90
  5.6 CIML Elements Definition 92
    5.6.1 Component Declaration 93
    5.6.2 Initialize Declaration 95
    5.6.3 Interaction Declaration 97
  5.7 Implementation of CIML Specification 99
  5.8 The CIML Schema Diagram 99
  5.9 The CIML XML Schema Definition 101
  5.10 CIML and Mediator Connector 103
  5.11 Summary 103
# LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1: Taxonomy based on composition (Lau and Wang, 2005)</td>
<td>7</td>
</tr>
<tr>
<td>1.2: Existing component models and their composition schemes</td>
<td>8</td>
</tr>
<tr>
<td>2.1: Some types of scripting languages in different domains</td>
<td>17</td>
</tr>
<tr>
<td>3.1: Example ADLs</td>
<td>68</td>
</tr>
<tr>
<td>3.2: Summary of related work</td>
<td>76</td>
</tr>
<tr>
<td>4.1: Criteria based on number of components and total CBO</td>
<td>81</td>
</tr>
<tr>
<td>5.1: XML Markup Types</td>
<td>87</td>
</tr>
<tr>
<td>5.2: Use of XML in Other Scientific and Business Fields</td>
<td>89</td>
</tr>
<tr>
<td>5.3: CIML Syntax</td>
<td>91</td>
</tr>
<tr>
<td>5.4: CIML elements (tags)</td>
<td>93</td>
</tr>
<tr>
<td>5.5: Element descriptions for components declaration</td>
<td>93</td>
</tr>
<tr>
<td>5.6: Element descriptions for initial method calls</td>
<td>95</td>
</tr>
<tr>
<td>5.7: Element description for an interaction</td>
<td>97</td>
</tr>
<tr>
<td>6.1: Description of actors in the use case diagram</td>
<td>110</td>
</tr>
<tr>
<td>7.1: Measurement values of CBO components in AJcFgraph Builder Tool</td>
<td>165</td>
</tr>
<tr>
<td>7.2: Measurement values of CBO components in simple Bank system</td>
<td>172</td>
</tr>
<tr>
<td>7.3: Measurement values of CBO components in MP3 player system</td>
<td>179</td>
</tr>
<tr>
<td>7.4: Measurement values of CBO components in Display Map system</td>
<td>187</td>
</tr>
<tr>
<td>7.5: Measurement values of CBO applied on components in the Bank system</td>
<td>190</td>
</tr>
<tr>
<td>7.6: Measurement values chart of CBO applied on components in DisplayMap system</td>
<td>190</td>
</tr>
</tbody>
</table>
7.7: Measurement value of CBO using exogenous connectors compared with mediator connector
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1: Direct Message Passing</td>
<td>5</td>
</tr>
<tr>
<td>1.2: Indirect Message Passing</td>
<td>6</td>
</tr>
<tr>
<td>1.3: Research activities flowchart</td>
<td>12</td>
</tr>
<tr>
<td>2.1: An idealized component life cycle (Lau, Wang, 2007)</td>
<td>17</td>
</tr>
<tr>
<td>2.2: Design Pattern Space (Gamma et al. 1994)</td>
<td>26</td>
</tr>
<tr>
<td>2.3: A bank system using Mediator design patterns</td>
<td>30</td>
</tr>
<tr>
<td>3.1: The Enterprise JavaBean Architecture</td>
<td>37</td>
</tr>
<tr>
<td>3.2: CORBA Component</td>
<td>42</td>
</tr>
<tr>
<td>3.3: CCM Container</td>
<td>43</td>
</tr>
<tr>
<td>3.4: A request passing from client to the object implementation (Hamilton, 1997)</td>
<td>44</td>
</tr>
<tr>
<td>3.5: CosNaming IDL</td>
<td>46</td>
</tr>
<tr>
<td>3.6: Interoperability by ORB-to-ORB communication (Hamilton, 1997)</td>
<td>46</td>
</tr>
<tr>
<td>3.7: Using services provided by a COM object through interface pointer</td>
<td>49</td>
</tr>
<tr>
<td>3.8: A Clock COM object</td>
<td>50</td>
</tr>
<tr>
<td>3.9: Three methods for accessing COM objects (Microsoft, 1995)</td>
<td>51</td>
</tr>
<tr>
<td>3.10: Cross-process communication in COM(Microsoft, 1995)</td>
<td>53</td>
</tr>
<tr>
<td>3.11: Structure of proxy in COM (Microsoft MSDN, 2007)</td>
<td>54</td>
</tr>
<tr>
<td>3.12: Structure of the stub in COM (Microsoft MSDN, 2007)</td>
<td>55</td>
</tr>
<tr>
<td>3.13: An interface definition</td>
<td>59</td>
</tr>
<tr>
<td>3.14: A component definition</td>
<td>59</td>
</tr>
<tr>
<td>3.15: A Koala Compound Component (Ommering, 2000)</td>
<td>60</td>
</tr>
</tbody>
</table>
3.16: A configuration definition

3.17: A sample declaration of a SOFA component in CDL

3.18: An example of an interface and frame protocols

3.19: A simple example of a client server with a single client represented in Acme

3.20 A simple specification in Rapide

3.21: A simple specification in Wright

3.22: Architecture of a bank example using exogenous connector (Lau et al., 2005)

4.1: Research Methodology Overview

5.1: Components (Objects) in an interaction diagram.

5.2: Initializing in an interaction diagram

5.3: Interaction description part

5.4: XML Schema for CIML components composition structure description

5.5: The CIML XML Schema Definition

6.1: The Process of building up a system using CIML and mediator connector

6.2: Use case diagram of mediator connector

6.3: Collaboration for the Parse CIML document

6.4: Communication diagram for Parse CIML document

6.5: Collaboration diagram for Create Interaction

6.6: Communication diagram for Create Interaction

6.7: Collaboration diagram for Create ComWrapper

6.8: Communication diagram for Create ComWrapper

6.9: Collaboration diagram for Create MethodCall

6.10: Communication diagram for Create MethodCall
6.34: Constructor of the MethodCall class

6.35: Constructor of the Parameter Class

6.36: Part of constructor code of mediator connector

6.37: Part of the code for a system using mediator connector

6.38: Invoking startParser method by mediator connector

6.39: Constructor of CIMLParse

7.1: Measurement framework

7.2: AJcFgraph application

7.3: AJcFgraph: measurement result before composition

7.4: AJcFgraph: class diagram before composition

7.5: AJcFgraph: measurement result after composition without MC

7.6: AJcFgraph: class diagram not using mediator connector

7.7: AJcFgraph: measurement result after composition using MC

7.8: AJcFgraph: class diagram using mediator connector

7.9: Measurement chart of CBO applied on components in AJcFgraph tool

7.10: Bank application

7.11: Bank: measurement result before composition

7.12: Bank: class diagram before composition

7.13: Bank: measurement result after composition without MC

7.14: Bank: class diagram without MC

7.15: Bank: measurement result after composition using MC

7.16: Bank: class diagram using MC

7.17: Measurement chart of CBO applied on components in Bank system

7.18: MP3Player application
7.19: MP3Player: measurement result before composition
7.20: Mp3Player: class diagram before composition
7.21: MP3Player: measurement result after composition without MC
7.22: MP3Player: class diagram without MC
7.23: MP3Player: measurement result after composition using MC
7.24: MP3Player: class diagram using MC
7.25: Measurement chart of CBO applied on components in MP3 Player
7.26: Find and show the address assigned to a phone number
7.27: DisplayMap: measurement result before composition
7.28: DisplayMap: class diagram before composition
7.29: DisplayMap: measurement result after composition without using MC
7.30: DisplayMap: class diagram after composition without MC
7.31: DisplayMap: measurement result before composition
7.32: DisplayMap: class diagram using MC
7.33: Measurement chart of CBO applied on components in Display Map system
7.34: Architecture of a bank example using exogenous connectors (Lau et al., 05)
7.35: Architecture of a bank example using mediator connector
7.36: Outline code of a bank system using exogenous connectors (Lau et al., 05)
<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>API</td>
<td>Application Programming Interface</td>
</tr>
<tr>
<td>CIML</td>
<td>Component Interaction Markup Language</td>
</tr>
<tr>
<td>CBSD</td>
<td>Component-Based Software Development</td>
</tr>
<tr>
<td>GUI</td>
<td>Graphical User Interface</td>
</tr>
<tr>
<td>JAXP</td>
<td>Java API for XML Processing</td>
</tr>
<tr>
<td>JDOM</td>
<td>Java Document Object Model</td>
</tr>
<tr>
<td>JVM</td>
<td>Java Virtual Machine</td>
</tr>
<tr>
<td>SAX</td>
<td>Simple API for XML</td>
</tr>
<tr>
<td>W3C</td>
<td>World Wide Web Consortium</td>
</tr>
<tr>
<td>XML</td>
<td>eXtensible Markup Language</td>
</tr>
<tr>
<td>XSD</td>
<td>XML Schema Definition</td>
</tr>
<tr>
<td>XSLT</td>
<td>eXtensible Stylesheet Language Transformation</td>
</tr>
</tbody>
</table>
CHAPTER 1

INTRODUCTION

1.1 Background

The design process in most engineering disciplines is based on reuse of existing systems or components. Mechanical or electrical engineers do not normally specify a design where every component has to be manufactured specially. They base their design on components that have been tried and tested in other systems.

Reuse-based software engineering can be compared to software engineering strategy where the development process is geared to reusing existing software. Although the benefits of reuse have been recognized for many years (Mellor, 1968), it is only the past 10 years that there has been an evolution from original software development to reuse-based development.

Many techniques have been developed to support software reuse during the past 20 years (Sommerville, 2004). Reuse is possible at different levels (from simple function to complete application). There are a number of ways to support software reuse:

- Legacy system wrapping
- Design patterns
- Program libraries
• Configurable vertical application
• Components Off The Shelf (COTS) integration
• Application product lines
• Program generators
• Component frameworks
• Component-based Software Engineering (CBSE)
• Aspect-oriented software development
• Service-oriented systems

Component-based Software Engineering (CBSE) emerged in the late 1990s as a reuse-based approach to software system development. CBSE is the process of defining, implementing and integrating or composing the loosely coupled and independent developed components into system.

The foundation of any Component-Based Software Development (CBSD) methodology is its underlying component model, which defines what components are, how they can be constructed, how they can be composed or assembled. Components communicate with their environment only through their interfaces, so it is the interface which provides all the information needed.

1.1.1 Composition

Generally composition means taking two or more constructs and putting them together in some way. Composition is a constructive operation—its result is a new thing that has