ENHANCEMENT OF RAPID OBJECT PROCESS FOR EMBEDDED SYSTEM (ROPES) TO SUPPORT PATTERN ORIENTED DEVELOPMENT

MOHD ADHAM BIN ISA

A project report submitted in partial fulfilment of the
requirements for the award of the degree of
Master of Science (Computer Science)

Faculty of Computer Science and Information Systems
University of Technology Malaysia

OCTOBER 2009

ACKNOWLEDGEMENT

In preparing this project report, I wish to express my sincere appreciation to my supervisor Dr. Dayang Norhayati Binti Abang Jawawi for the guidance, advice and encouragement during my studying. The support and suggestion that Dr. Dayang gives inspired me to going through in this project.

I would like to thanks to Software Engineering Lab members in Universiti Teknologi Malaysia for their helps and supports.

Finally my special thanks to my parent for their love and care especially my beloved wife Mastura Md. Hassan and my beautiful daughter Elya Darwisyah for their support and cheering me up at those difficult time.

ABSTRACT

The complexity of Embedded Real Time (ERT) software development represents a challenging of analysing, designing and building ERT software. From this standpoint, the complexity of ERT software development means a challenging to adapt all ERT software requirements such as timing and resource constraints into its software lifecycle. Against these claims, a wide range of software development methodologies have been devised such as patterns. Patterns codify an effective solution for recurring problems that allows software engineers to reuse. By applying patterns into ERT software development, the complexity of ERT software development can be decreased and at the same time promote high degree of reuse through software patterns. In this project, the integrated Rapid Object Process for Embedded System (ROPES) and Pattern-oriented Analysis and Design (POAD) methodology has been developed to represent a promising way to build ERT software with software patterns reuse. To make the integrated methodology more compelling and confirm the rules of patterns oriented modelling, the integrated ROPES and POAD metamodel has been developed. The aim of the integrated metamodel is to conform the correctness of the integrated methodology modelling rules in term of pattern uses. In addition, the integrated ROPES and POAD software process also has been built as the continuity to describe concrete integrated software development process. To verify the correctness of the integrated metamodel, the mapping process of Meta-Object Facility (MOF) using graph theory has been conducted. The results of implementing the integrated metamodel and software process for Feedback Control System (FCS) shows that the complexity of ERT software development has been decreased besides promote software patterns reuse.

ABSTRAK

Pembangunan perisian Sistem Masa Nyata Terbenam (ERT) yang kompleks menunjukkan keunikan menganalisa, merekabentuk dan membina perisian ERT. Daripada ciri-ciri ini, pembangunan perisian ERT yang kompleks bermakna cabaran untuk merealisasikan semua keperluan perisian ERT seperti masa and kekekangan sumber keatas proses kitaran pembangunan perisiannya. Berpandukan kenyataan tersebut, pelbagai metodologi pembangunan perisian telah diubah atau dinaiktaraf sebagai contoh adalah corak. Corak memberikan penyelesaian yang efektif bagi penyelesain masalah yang berulang serta membolehkan jurutera perisian menggunakan semula kaedah ini bagi menyelesaikan masalah lain. Dengan mengaplikasikan penggunaan corak keatas pembangunan perisian ERT, kerumitan prosesnya dapat dikurangkan selain menggalakkan penggunaan semula corak. Di dalam projek ini, gabungan *Rapid Object Process for Embedded System (ROPES)* dan *Pattern-oriented Analysis and Design (POAD)* metodologi telah dibangunkan. Untuk menjadikan gabungan kedua-dua metodologi ini lebih kukuh dan mengikut peraturan pembangunan perisian berasaskan corak, gabungan metamodel telah dihasilkan. Matlamat utamanya adalah untuk memastikan rekabentuk perisian ERT mengikut corak. Selain itu, pembangunan proses perisian juga telah dibangunkan. Untuk memastikan gabungan metamodel tepat, proses teori graf bersama *Meta-Object Facility (MOF)* telah dilakukan. Hasil keputusan keatas pelaksanaan gabungan metamodel dan proses pembangunan perisian menunjukan kerumitan pembangunan perisian ERT telah dikurangkan dan penggunaan semula corak keatas rekabentuk perisian telah ditingkatkan.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

CBSE	Component Based Software Engineering
CBSD	Component Based Software Development
COMET	Concurrent Object Modelling and Architectural Design Method
DESS	Software Development Process for Real Time Embedded Software System
EJB	Enterprise Java Bean
ERT	Embedded Real Time
FCS	Feedback Control System
MARMOT	Method for Component-based Real Time Object Oriented Development and Testing
MOF	Meta Object Facility
OOA	Object Oriented Analysis
OOD	Object Oriented Design
OCL	Object Constraint Language
OMG	Object Management Group
PDMA	Pattern Driven Modelling and Analysis
PECOS	Pervasive Component System
POAD	Pattern Oriented Analysis and Design
POSA	Pattern Oriented Software Architecture
ROPES	Rapid Object Process for Embedded System
SOF	Special Octopus Feature
SPEM	Software Process Engineering Modelling
UML	Unified Modelling Language

LIST OF APPENDIX

CHAPTER 1

PROJECT OVERVIEW

1.1 Introduction

The necessity of the complex systems was felt with the demand from the large scale company or requirement. For the last fifty years, software systems have become more complex and larger. Following the traditional approaches which is developed a system from scratch eventually resulted the problems like failure to meet customer requirements, budget constraint and extend deadline. Then, in order to overcome those problems, it was realized that the software reusability is the essential factors that contributing of minimizing the existing problems. Software reusability allows parts of software component to be renewed or replaced by existing components. Thus, software reusability helps software developers to concentrate on adding more complex system functionalities rather than focusing on developing basic components. Here the concept of software reusability was constructed which eventually creating the software development methodology called Component Based Software Engineering (CBSE).

Component Based Software Engineering (CBSE) is the methodology that ensuring the improvement of the software quality, increasing software productivity, effective management for complex software and wider range of usability (Crnkovic, 2003). CBSE main principles are building the systems from the existing and selecting components within the suitable software architecture. The software components are composed together with the specific glue code and composition technique. Szyperski (2002) defines a component is a unit of composition with specified interfaces and explicit context dependencies only. A software component can be deployed independently and is subjected to compose by a third party. Thus, a component must have interface and hiding information that enable to be composed each other. JavaBeans, EJB, COM, Koala, PECOS (Nierstransz et al., 2002) and ADLs (Clements, 1996) are the examples of component technologies. In addition, a component is designed to solve particular problems and resembles like a pattern that forces the developers to use predefined process or design to be used in software development process.

A pattern was introduced into the software development process as a means of manipulating hard-earned experience from the previous problems (Kotonya Hutchinson, 2005). Yacoub and Ammar (2004) defines a pattern as problems that frequently occurs in software development and then defines the explicit solution that it can be reused while Fernandez and Yuan (2000) defines a pattern is a recurring combination of meaningful units that occur in some context. Then, a pattern is enabling software engineer to capture the software requirements from early phases. Patterns can be divided into analysis pattern, design pattern and programming pattern. Analysis pattern occurs at the early phases and known as a group of concepts that representing a common construction in business modelling (Fowler, 1997). Design pattern is more matured than analysis pattern (Sesera, 2000). Design pattern is a design solution to a frequently recurring design problem in a given application domain (Yacoub *et al.,* 2000).

There are many software development methodology based on patterns were introduced such as POAD, PDMA, and Design pattern + CBSD. These methodologies use software patterns as its core specification. POAD is one of the methodologies that develop a system from the analysis phases until design phases by using software patterns. Thus, POAD is a structural approach of software development rather than behaviour approach (Yacoub and Ammar, 2004). In addition, POAD methodology leads into the bridging the gaps between software abstraction into software designs by combining software patterns artefacts into pattern-oriented software development. By the existing of various software patterns, there is a growing demand of searching a new and cost effective of software development paradigm. Then, one of the most promising solutions today is the pattern-oriented development.

A pattern-oriented development is an approach to deal with the complexity of software development. The complexity of the software development usually comes from the extra requirements such as timing and resource constraints. By applying pattern-oriented development in the software development process, the skeleton of the software can be perceived up-front by using knowledge of previously patterns. Moreover, it will give the traceability of software artefacts from the analysis to design. Moreover, there are a few advantages of using pattern in software development. The advantages are 1) the reuse experienced, 2)the development of standards types of solution, and 3)a normalized mechanism for abstracting form multiple example and method for categorizing problems (Kotonya and Hutchinson, 2005). The effectiveness of minimizing the complexity of software development can be realized by composing software patterns into the software development process.

The complexity of software development process is known as the most critical stage in the process of building software. Indeed, this is due to the complexity of the embedded real-time software requirements as a solution for ever more demanding applications required. ERT software requirements such as timing and resource

constraints represent a promising point of the complexity of ERT software product. Most of the embedded systems are characterized as real time systems, which consist of real time properties such as response time and worse case execution time (Crnkovic, 2005). Consequently, the complexity of the demanding ERT requirements will be likely is resulted the complexity of ERT software development.

The motivation of using software patterns in ERT software developments is twofold. Firstly, despite ERT software are being touted as a complex software due to its requirements, the most compelling form of ERT software would be to show how adopting software patterns can decrease the complexity of ERT software development to some standard of software process. This is clearly the major issue which this project seeks to address. Secondly, because of the implementation of software patterns can decrease the complexity of ERT software development to some degree that will have the major impact of the degree reuse of software patterns. To some extent, the high degree reuse of software patterns will decrease the complexity of the ERT software development. Moreover, this is envisioned that CBSE will be the suitable solution to inspire software reuse into ERT system with consideration of reusable, interoperable, up gradable and reliable features. Thus, patterns are promoting the precise planning and systematic software development process by using software pattern into ERT system development

1.2 Problem Background

Nowadays, the use of software in the different domains is highly demanded such as web application, windows environment as well as embedded real time system (ERT).

ERT system is a system that combines both hardware and software. The hardware component of the system tries to persuade the timing constraints while software reduces the overall cost and design flexibility (Jawawi, 2003). The interoperating of those elements affects the complexity of the software development. ERT software developments are crucial since the combination both hardware and software can lead into the complexity of the process. All of this events must have a systematic process in order to reduce the time and cost of development.

The design pattern is the defined solution of the hard-earned experience from the previous software development (Kotonya and Hutchinson, 2005). Thus, the component based is mechanism of software development which are the component is composed together in order to build the real time system (Zhang and Yunsheng, 2007). The uses of pattern and component together could reduce the complexity of the software process. This is because of the use of pattern for the whole process glued together with the component could reduce the time and cost of software development.

There a several methodologies that have been introduced focusing on component based and pattern in the software developments. Pattern oriented analysis and design (POAD) is a methodology that constraint on the uses of components and patterns from the analysis stages until the design (Yacoub and Ammar, 2004). However, this methodology only addresses to the general domain and does not specific for the ERT system. Rapid object-oriented process for embedded system (ROPES) focusing on the ERT software developments and capturing both system structural and behavioural elements (Douglass, 1999). However, ROPES does not address the use of patterns systematically in software development process. The patterns are occurs only at the design stages as well as does not have the systematic approaches in term of selecting and adopting the patterns. By considering the advantages and disadvantages of both methodologies, the combination of POAD and ROPES that focuses on ERT software

development could reduce the complexity of ERT software development and at the same time, can ensure the high degree reuse of software patterns.

1.3 Problem Statement

POAD methodology defined the process of analysis and design by using the selected pattern and consistently defined the pattern for the whole process. ROPES methodology defined the complex step of software development for ERT but there are no systematic approaches of using the pattern for each process. To promote the relevance of pattern-oriented for analysis and design level for the ERT domain, the consideration falls into ROPES and POAD methodologies. Instead of developing the new methodologies from the scratch, the hybrid methodologies for ROPES and POAD can ensure the use of pattern subject to the ERT domain. Therefore, this project will define the Metamodel and software process for the integrated ROPES and POAD. An integrated Metamodel for both methodologies represents the abstractions and semantics in system modelling in order to apply the use of patterns in ERT software development process and then constructed it in software process in order to get well-defined software development process. However, there are some problems need to be considered before the integrated POAD and ROPES metamodel can be realized. The problem arises are mapping process between both models, defining new development process for integrated metamodel and selecting a tool for supporting the integrated metamodel. All of this problems need to be solved in order to get suitable integrated metamodel and software process.

Therefore, this study proposes the integrated ROPES and POAD methodology to enable pattern oriented development for ERT domain. Hence, it may decrease the complexity of ERT software development and promote high degree of reuse in software patterns.

1.4 Project Aim

The aim of the project is to integrate POAD and ROPES methodologies. The motivation is to propose the use of patterns by using POAD methodology into the ERT software development in order to discover the weaknesses of the ROPES methodology. Indeed, ROPES weaknesses compromise non-systematic use of patterns in its software development lifecycle. Therefore, the integrated ROPES and POAD metamodel is proposed and consequently defines the software process for the integrated metamodel. To show the applicability of the proposed integrated methodology, the implementation of analysis and design into selected case study of small scale ERT system will be implemented by using selected modelling tool.

1.5 Objectives

The objectives of the study are:

i. To study and propose the metamodel for integrated POAD and ROPES methodologies.

ii. To define the software process for integrated POAD and ROPES Metamodel using Software Process Engineering Metamodel (SPEM).

iii. To demonstrate the applicability of the proposed process by using the I-Logix Rhapsody tool into selected case study.

1.6 Scopes

The scopes of the study are:

i. The analysis and design of Pattern oriented analysis and design (POAD) methodology.

ii. The analysis and design of Rapid Oriented Process for Embedded System (ROPES) methodology will only consider to demonstrates the pattern level analysis and modelling into the case study.

iii. The implementation only for small scale of embedded real time (ERT) system.

iv. The I-Logix Rhapsody tool is selected to support the integrated POAD and ROPES metamodel realization because I-Logix Rhapsody tool support UML 2.0. Focusing on application engineering and assume the components are already exists.

1.7 Significance of the project

The significance of this project is to promote the software pattern reuse in ERT software development. The study concentrates on a deep understanding of POAD and ROPES methodologies. Based on that knowledge, the integrated Metamodel will be introduced that supports the use of pattern on ROPES methodology. By providing the proposed Metamodel, the software process can be defined. The advantages that can be derived from this study are to motivate the use of pattern besides to enhance the ERT software development process in order to maximize the quality and minimize the cost of development.

1.8 Thesis Outline

Chapter 2 discusses the pattern-oriented development methodologies and ERT software development methodologies. The relationship between those methodologies with the uses of pattern and component in software development are reviewed.

In Chapter 3, the research methodology is conducted in achieving the project objectives and scopes. One case study is used that involving small scale ERT system.

Chapter 4 defines the construction of the integated ROPES and POAD metamodel and validation process of the proposed metamodel by using Meta-Object Facility (MOF) and graph theory.

Chapter 5 describes the definition of the integrated ROPES and POAD software process into the formal form to enrich the systematic way of system development process.

Chapter 6 purposes are to prove the applicability of the integrated ROPES and POAD metamodel and software process. The proven process of proposed software process heavily relied on case study. The Feedback Control System was selected to be used for the proven process.

REFERENCES

Angelov, C., Sierszecki, K., & Marian, N. (2008). Component-Based Design of Embedded Software: An Analysis of Design Issues. In *Lecture Notes in Computer Science* (pp. 1-11). Berlin: Springer Berlin / Heidelberg.

Baelen, S. V., Gorinsik, J., & Wills, A. (2001). The DESS Methodology. *Deliverable D.1 Version 01 - Public* .

Buschman, F., Meunier, R., Rohnert, H., & Sommerlab, P. (1996). *Pattern Oriented Software Architecture:A System of Patterns.* John Wiley and Sons.

Bunse, C., Gross, H. G., & Peper, C. (2008). Embedded System Construction – Evaluation of Model-Driven and Component-Based Development Approaches. *MODELS`08 Workshop ESMDE* .

Clements, P. (1996). A Survey of Architecture Description Languages. *In Proceedings of the Eighth International Workshop on Software Specification and Design* (pp. 16-25). Paderborn, Germany: ACM.

Coad, P. (1997). *Object Models:Strategies, Patterns and Applications.* Yourdon Press.

Crankovic, I. (2003). Component based Software Engineering- New Challenges in Software Development. *25th International Conference on Information Technology Interfaces*, (pp. 9-18).

Crnkovic, I. (2005). Component-based Software Engineering for Embedded Systems. *ICSE'05 ACM* .

Crnkovic, I., Stig, L., & Michel, C. (2005). Component Based Development Process and Component Lifecycle. *Journal of Computing and Information technology* , 321-327.

Demicheil, L., Yalcinalp, L., & Krishnan, S. (2001). Enterprise JavaBeans Specification Version 2.0.

Domiczi, E., Farfarakis, R., & Jürgen, Z. (2000). Octopus/UML Artifact Reference Version 1.3.

Domiczi, E., Rallis, F., & Jürgen, Z. (2000). Release of Octopus/UML. *Nokia Research Center* .

Dong, J., Alencar, P. S., & Cowan, D. D. (2004). A behavioral analysis and verification approach to pattern-based design composition. *Software and Systems Modeling* , 262-272.

Douglass, B. P. (1999). *Doing Hard Time : Developing Real-Time System Using UML Objects, Framework and Patterns.* Addison-Wesley.

Felix, L., & Narashima, B. (2003). Object-Oriented Analysis using Patterns: A Review and Research Opportunities.

Fernandez, E. B., & Yuan, X. (2000). Semantic Analysis Patterns.

Fernando, L. C. (2005). A New Component Based Life Cycle Model. *Journal of Computer science* , 76-82.

Finkelstein, A. (2000). The Future of Software Engineering. *Proceedings of 22nd International Conference on Software Engineering* , ACM Press.

Fowler, M. (1997). *Analysis Patterns, Reusable Object Models.* Addison Wesley.

Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (1995). *Design Patterns, Elements of Reusable Object Oriented Software.* Addison-Wesley.

Garlan, D., & Shaw, M. (1996). *Software Architecture: Perspectives on an Emerging Discipline.* Prentice Hall.

Gomaa, H. (2001). *Designing Concurrent, Distributed and Real Time Applications with UML.* Addison-Wesley.

Harb, D., Bouhours, C., & Leblanc, H. (2009). Using an Ontology to Suggest Software Design Patterns Integration. In *MODELS 2008 Workshops, LNCS 5421* (pp. 318-331). Berlin: Springer-Verlag Berlin Heidelberg.

Heinenam, G., & Councill, W. (2001). *Component Based Software Engineering.* Addison-Wesley.

Jawawi, D. (2006). *A Framework for Component-Based Reuse for Autonomous Mobile Robot Software.* PhD Thesis: Universiti Teknologi Malaysia.

Jawawi, D. (2003). *Embedded Real Time System:Technical Report.* Universiti Teknologi Malaysia.

Jawawi, D., Mohamad, R., Deris, S., & Mamat, R. (2005). Transforming Pattern-Oriented Models into Component-Based Models for Embedded Real Time Software Development. *MySEC'05* .

Jerry, O. (2007). *Pattern Oriented Analysis and Design Theory.*

Konrad, S., & Cheng, B. H. (2005). Real-time Specification Patterns. *Proceedings of the 27th International Conference of Software Engineering (ICSE05)* .

Kotonya, G., & Hutchinson, J. (2005). Pattern and Component Oriented System Development. *Proceeding of the 2005 31st EUROMICRO Conference on Software Engineering and Advanced Applications* .

Lau, K. K., & Wang, Z. (2005). A Taxonomy of Software Component Models. *Proceeding of the 2005 31st EUROMICRO Conference on Software Engineering and Advanced Applications* .

Luigi, L., & Bianco, V. d. (2000). Software Development Process for Real-Time Embedded Software Systems (DESS). *Deliverable D5.2/D5.3/D5.4 DESS process modeling:methodology and support* .

Meyer, B. (2003). The grand Challenge of Trusted Components. *Proceedings ICSE 2003 IEEE* , 660-667.

Natale, M. D. (2008). Design and Development of Component-Based Embedded Systems for Automotive Applications. In *Lecture Notes in Computer Science, Reliable Software Technologies – Ada-Europe 2008* (pp. 15-29). Berlin: Springer Berlin / Heidelberg.

Nicola, J., Mayfield, M., & Abney, M. (2001). *Streamlined Object Modelling: Patterns, Rules and Implementation.* Upper Saddle River, N.J: Prentice Hall.

Nierstrasz, O., Grenssler, T., & Schonhage, B. (2002). Components for Embedded Software. *The PECOS Approach CASES* .

OMG. (2007). *UML Superstructure Specification.* http://www.omg.org.

Rajasree, M. S., Jithendra, P., Reddy, K., & Janakiram, D. (2001). *Pattern Oriented Software Development: Moving Seamlessly from Requirements to Architecture.*

Riehle, D., & Zullighoven, H. (1996). Understanding and Using Patterns in Software Development. *Theory and Practice of Object System* , Vol, 2(1) 13-33.

Sesera, L. (2000). Analysis Pattern. *SOFSEM'2000 Lecture Notes in Computer Science Series*

Sesera, L. (2001). Hierarchical Patterns:A Way to Organize Analysis Patterns. *SYSTEMATICS, CYBERNATICS AND INFORMATICS* , Volume 3.

Sommerville, L. (2004). *Software Engineering 7th Edition.* Addison-Wesley.

Staines, A. S. (2007). A Comparison of Software Analysis and Design Methods for Real Time Systems. *Proceedings of World Academy of Science, Engineering and Technology*, (p. Volume 21).

Szyperski, C. (2002). *Component Software: Beyong Object Oriented Programming.* Addison-Wesley.

Tichy, M. (2006). Pattern-Based Synthesis of Fault-Tolerance Embedded Systems. *SIGSOFT'06/FSE14 ACM* .

Wan Zulkifeli, W. R. (2008). *Kajian Penggunaan Corak Rekabentuk Masa Nyata Dalam Pembangunan Sistem Masa Nyata Berasaskan Metodologi Pattern Oriented Analysis and Design (POAD) dan Rapid Object Process for Embedded System (ROPES).* Malaysia.

Yacoub, S. M., Xue, H., & Ammar, H. H. (2000). Automating the Development of Pattern Oriented Designs for Application Specific Software Systems. *Proceeding IEEE* .

Yacoub, S., & Ammar, H. H. (2004). *Pattern Oriented Analysis and Design:Composing Patterns to Design Software System.* Addison Wesley.

Yau, S., & Dong, N. (2000). Integration in Component based Software Development Using Design Pattern. *Proceeding IEEE* .

Zhang, J. Z., & Yunsheng. (2007). Component-oriented Modeling and Design of Hierarchical Hybrid Control System. *IEEE International Conference on Control and Automation Guangzhou, CHINA* .