

Acknowledgement

I would like to express my deepest gratitude to all the professors for their support and guidance .

*A very special and well-deserved thank to **Dr. Nisreen Beshir**. As my supervisor, She helped me to successfully coordinate my ideas, offered direction with the implementation of those ideas, provided insight in the interpretation of my work, and supported me .I thank **Dr. Nael Salman** As one of my professors, he first introduced me to the concept of Software Engineering.*

I would also like to thank My family and friends. Thank you Software professionals around the world. Sincere thanks to the individuals that participated in my research .

Abstract

During software evolution, programmers spend time and effort in the comprehension of program and that is due to the fact that the documentation is often incomplete, inconsistent and outdated. In order to avoid these problems, software could be re-documented.

Software re-documentation enables the understanding of software that aids the support, maintenance and evolution of software. Re-documentation is implemented by different approaches. Reverse Engineering is one of these approaches that provide a better understanding of an existing system by maintainers and developers, especially when faced by a large and evolving legacy system.

This study proposes a framework for systems re-documentation based on reverse engineering approach. The re-documentation is done using a reverse engineering tool that generates graphical representations of a system which is then used to produce documentation in a form of a standard document UML notation. Since the quality of the generated documentation is important for program understanding and software evolutions, the study also proposes a model for evaluating the quality of the generated documentation.

أثناء تطوير البرمجيات يصرف المبرمجين الوقت والجهد في فهم البرنامج وذلك بسبب أن التوثيق قديم ومتناقض وناقص في اغلب الأحيان . إعادة التوثيق تساعد في تفادي مثل هذه المشكلات . يمكن إعادة توثيق البرمجيات من فهم البرنامج الذي يساعد الدعم والصيانة والتطوير للبرمجيات .

إعادة التوثيق تُطبّق بمنهجيات مختلفة . الهندسة العكسية احد هذه المنهجيات التي تزودنا بادراك أفضل للنظام من قبل فرق الصيانة والتطوير ، خاصة عند تعاملهم مع الأنظمة القديمة والكبيرة .

تقترح هذه الدراسة إطار لإعادة التوثيق مستندة على نظرية الهندسة العكسية . إعادة التوثيق تُطبّق باستخدام احد أدوات الهندسة العكسية والتي تولّد تمثيل رسومي للنظام والذي يستخدم لإنتاج وثيقة قياسية في صيغة ترقيم UML . وبما أن جودة التوثيق المؤلّد مهمة لفهم البرنامج وتطويره ، تقترح الدراسة أيضا نموذج لتقييم جودة التوثيق المؤلّد .

Table of Contents

TITLE	P.NO
Acknowledgements	i
Abstract	ii
الخلاصة.....	iii
Table of Contents	iv
List of Figures.....	vi
List of Table	vi
List of Abbreviations.....	vii
1 Ch 1: Introduction	
1.1 Introduction	1
1.2 Statement of the Problem	2
1.3 Motivation and Objectives	2
1.4 Research steps	4
1.5 Research organization	5
2 Ch 2 : literature review	
2.1 : Overview	6
2.2 : Software Re-documentation	7
2.2.1 : Re-documentation definition	7
2.2.2 : Goals of the software Re-documentation	8
2.2.3 : Re-documentation Categorization	9
2.2.4 : Re-documentation Process	10
2.2.5 : Re-documentation Approaches	11
2.3 : Reverse Engineering	13
2.3.1 : RE Definition	14
2.3.2 : RE Tasks	14
2.4 : Quality of Documentation	16
2.4.1 : Quality definition	17
2.4.2 : Measuring the Quality	17
3 Ch 3 : Research Methodology	
3.1 Introduction.....	21
3.2 The Target SQL/DB system : ShM.....	22
3.3 Overview to Rigi Reverse Engineering Tool	22
3.4 Re-documentation using Rigi tool	23
3.4.1 : Read the source code	23
3.4.2 : Load initial Graph , the Root window	23
3.4.3 : Visualization , The Graphical Representation	24
3.4.4 : The Textual Representation	26
3.5 : Generating Documents	27
3.6 : Proposed Documentation Quality Model	31

4 Ch 4: Results and Discussion	
4.1 : Results	33
4.2 : Discussion	34
5 Ch 5 : Conclusion and Recommendations	
5.1 : Conclusion	35
5.2 : Recommendations and Future work	36
Appendix A	37
Appendix B	39
References	42

List of Figures

NO	TITLE	P.NO
1.1	: Research steps.....	4
2.1	: A Classification of software re-documentation.....	9
2.2	: Overview of the re-documentation process.....	10
2.3	: Difference between Forward and Reverse engineering	13
2.4	: Reverse engineering Tasks.....	14
2.5	: Documentation quality meta-model overview.....	19
3.1	: Research Methodology steps	21
3.2	: The main windows of RigiEdit.....	23
3.3	: The Graphical Representation.....	24
3.4	: View information from selected node	26
3.5	a :ShM class diagram.....	28
	b :ShM class diagram with details.....	29
3.6	Component ports interface for services.....	30
3.7	: Documentation Quality Model.....	31

List of Tables

NO	TITLE	P.NO
Table 2.1	: KPIs Framework	20
Table 3.1	:Class diagram construct Vs Rigi graph construct	27
Table 4.1	: Documents Evolution Result	33

List of Abbreviations

RE.	Reverse Engineering
DQM.	Document Quality Model
MOR.	Model Oriented Re-documentation
TSs.	Technological Spaces
KPIs.	Key Performance Indicators framework
RSF.	Rigi Standard Format