Al-Azhar University,
Faculty of Science (Girls),
Department of Mathematics.

Improvement of Formal Methods Approach Using Pattern Perspective in Real Time System Applications

A thesis submitted for

Ph.D. Degree

in

Computer Science

by

Enas El-Sayed Mohamed El-Sharawy

Department of Mathematics, Faculty of Science (Girls),
Al-Azhar University.

Under Supervision of:

Prof. Dr.
Ahmed Kamel El-Khouly
Al-Azhar University,
Faculty of Science (Girls),
Department of Mathematics,
(Mathematics section)

Prof. Dr.
Gaber Ahmed El-Sharawy
Al-Azhar University,
Faculty of Science (Girls),
Department of Mathematics,
(Computer science section)

Dr.
Eman Karam El-Sayed
Al-Azhar University,
Faculty of Science (Girls),
Department of Mathematics,
(Computer science section)

Cairo, 2014

ACKNOWLEDGEMENT

Prayerful thanks, at first to our Merciful God who gives me everything I have.

I wish to express my sincere gratitude and deepest thanks to Prof. Dr.A.K El-kholy and G.A. El Sharawy, for their active supervision, and constructive criticism during the progress of this thesis.

I'm greatly indebted to Dr. Eman Karam El-Sayed for providing me most of the Guidance needed for the completion of this work.

My thanks also go to the Department of Mathematics and faculty members; for their help and support.

My vocabulary utterly fails in expressing my accolade to my revered parents who brought me to this stage. I fall short of words for the moral support extended by them. I also gratefully acknowledge the help and moral support rendered by all family members for perusing further studies.

My thanks also extended to my husband and my sweet daughter, for their continuous encouragement, their patience, interest and helpful cooperation which made this investigation possible.

Thank giving is not merely a customary word; I realize that it is of immense significance too.

Grateful and indebted

(Enas El-Sharawy)

Abstract

The systematic approach (formal method) is a correct valid path to convert software requirement to executable code. Each Formal Method with Proofs FMP have theorem behind it as B theorem with Event-B formal method. By formal method we can build large computerized complex systems. That is by three techniques: refinement, decomposition and generic instantiation. In this thesis, we integrate refinement and generic instantiation techniques in Event-B formal method. Practically, we used RODIN platform and its database tools of the Static Checker, the Proof Obligation Generator, the Prover and the Translator from B model to a programming language.

The main aim of this work is rising the degree of automation and increasing efficiency of theorem proving technique to decrease the cost of such developments depends directly on them. And also increase correctness and validation degree. All of them with stable reusability. So we propose using automatic theorem provers known as SMT-solvers with Event-B pattern. The benefit of this approach is to have strong validation pattern. This approach enhances the pattern reusability with high degree of automatic proving and low effort. Then, we propose a new refinement of ATM UML model by agentless data collection approach and convert it to be Event-B pattern. That convert is more suitable for mathematician. That convert reduce the proof obligation.

Finally, we discuss a logical correct path to translate Event-B pattern to C# language code. That is for both models and algorithms.

Keywords: Formal method, Event-B, Proof Obligations, UML-B Pattern, Event-B Pattern, ATM model, SMT-Solver, agentless data collection approach, Code Generation.

Contents

LIST OF FIGURES

Figure No. **page**

Figure No. **page**

Figure No. **page**

LIST OF TABLES

LIST OF CHARTS

Chart **Page**

List of Abbreviations

FM	Formal Method
FMP	Formal Method with Proofs
RODIN	Rigorous Open Development environment for complex systems INdustry day
VDM	Vienna Development Method
CSP	Communicating Sequential Processes
KLOC	1000 Line Of Code
PO	Proof Obligation
SMT-SOLVER	Satisfiability Modulo Theory- Solver
INV	Invariant Preservation Proof Obligation Rule
FIS	Feasibility Proof Obligation Rule:
GRD	Guard Strengthening Proof Obligation Rule
MRG	The Guard Merging Proof Obligation Rule
SIM	Simulation Proof Obligation Rule
NAT	The Numeric Variant Proof Obligation Rule
FIN	The Finite Set Variant Proof Obligation Rule
VAR	The Variant Proof Obligation Rule
WFIS	The Non-Deterministic Witness Proof Obligation Rule
THM	The Theorem Proof Obligation Rule
BSA	Bank Service Agent
ATM	Auto Teller Machine
UML-B	Unified Modelling Language - B
EB2C#	Event-B to C#
API	Application Programming Interface

Chapter 1: Introduction

1.1 Introduction

Formal Methods are mathematically based modeling techniques used to specify and verify hardware and software systems. Z method, VDM (Vienna Development Method), B-Method (also known as classical B) and Event-B are among the most recent formal methods [5, 8].

Event-B is an evolution of the classical B. Event-B uses the concept of Refinement in modeling [31]. Event-B modeling starts with an abstract specification of a system. Details are added during refinement steps in order to arrive at a more detailed model. The mathematical language of Event-B is based on set theory and first order logic. Based on the Event-B language, a set of proofs can be produced and discharged for each Event-B model. RODIN is an open source, extensible and integrated modeling tool supporting Event-B. This tool is not only used as a modeling environment, but also provides an integrated environment for proving properties of models [9, 6, 26, 59].

Formal modelling is not only constructing descriptions, but also proving some properties about the formal models. RODIN provides an integrated environment for both modelling and proving. Extensibility of RODIN makes it easy for new features to be added to it. During recent years, some Eclipse based plug-ins were developed and added to RODIN: SMT-Solver as prover, ProB [71] as an animator, UML-B [72] as graphical environment provider, B2C# as a translator from B to C# code and, B2Latex [70] as a translator from B to Latex, have been developed and added to RODIN. Recently Event-B has been applied to developing industrial cases. However building models of large and complex systems results in large and complex models and difficult proofs. Some techniques

such as Pattern approach, SMT-Solver, UML and Event-B can help to solve this difficulty.

This thesis focuses on pattern as an approach for modeling and proving large and complex systems using Event-B. This approach enables developers to reuse the proved model in Event-B. Refinement in Event-B is too general. It does not explicitly show all of the relations between behaviors of the abstract model, called abstract events, and the behaviors of the refinement, called the concrete events. This approach is also capable of showing an overall structure of several refinement levels. Therefore it provides an effective way to handle complex development.

On the other hand, providing patterns and UML-B, makes the modelling of large systems more manageable. Using UML-B and patterns, we can achieve reusability and visualization in Event-B development.

Finally, we use code generation approach to generate C# code from Event-B model. When we do this conversion, we achieve the validation of C# code and guarantee that the code fulfills the requirement of model.To achieve reusability, correctness and visualization, the used tool has been developed as a plug-in tool for the RODIN platform.

1.2 Thesis Motivation and Contribution

The intent of this work is to give some insights on modeling and formal modeling. These activities are supposed to be performed *before* undertaking the effective coding of a computer system, so that the system in question will be correct by construction.

We will also understand how it is possible to detect the presence of inconsistencies in our models just by the fact that some proofs cannot be done. The failure of the proof will provide us with some helpful clues about what is wrong or insufficiently defined in our model. We will use