POLICY-BASED MANAGERS COORDINATION FOR SELF-ADAPTIVE SOFTWARE APPROACH

ABDELHAMID ABDELHADI MANSOR HAMED

A thesis submitted in fulfilment of the requirements for the award of the degree of Doctor of Philosophy (Computer Science)

Faculty of Computing
Universiti Teknologi Malaysia

JUNE 2013
To

My Parents
My Wife, Hana
My Mother-in-Law
My Sons, Ghassan and Safwan
First and forever, I would like to thank God, for blessing me in my life, my study and in preparing my thesis. Then, I would like to express my sincere gratitude to my advisor Assoc. Prof. Dr. Wan Mohd Nasir Wan Kadir, for the continuous support of my Ph.D study and research, for his patience, motivation, enthusiasm, and immense knowledge. His guidance helped me in all the time of research and writing of this thesis. I could not have imagined having a better advisor and mentor for my Ph.D study.

Besides my advisor, I would like to thank my co-supervisor, Assoc. Prof. Dr. Toni Anwar, who has supported me throughout my thesis with his patience and knowledge.

Moreover, I would like to thank the examiners of my Ph.D thesis for their invaluable comments on the thesis. I am also in debt to Universiti Teknologi Malaysia (UTM) for their financial support under Research University Grant Scheme (RUGS).

Finally, I thank my wife Hana and my sons Ghassan and Safwan, to whom this thesis is dedicated, for their endless supports during my stay in Malaysia. They gave me courage and the strength to never give up. I also thank my parents and my family in law for their encouragement and support.
ABSTRACT

Due to the complexity and rapidly changing environment, it is crucial for a software system to be adaptable. One of the key approaches to achieve adaptability is policy-based management. Policy-based management has been acknowledged mostly as a methodology that provides flexibility, scalability, adaptability, and support to assign system resources, control, quality of service, and security by considering administratively specified rules. The promotion of policy-based management was to commit to these features at runtime because of changeable concurrent system conditions ensuing from the interactions of users' applications and existing resources. This thesis proposes an adaptive policy-based management approach called Policy-based Manager Coordination (PobMC) based on Event-Condition-Action (ECA). The aim of PobMC approach is to deal with the critical nature of avoiding the policy conflict problem. This approach facilitates policy conflict avoidance and static analysis to address the inconsistencies of multiple manager and modality conflicts when two or more policies are enforced simultaneously. The PobMC will also coordinate managerial tasks when multiple rules are simultaneously triggered. The mobility of devices and applications in complex system complicates policy design. Rules must be added or revoked when the composition of a system changes. Static and dynamic analysis algorithms are proposed; moreover, the need for these algorithms on various complex systems and their performance evaluation is demonstrated. In this thesis, a modelling of PobMC based on an ECA framework is presented. The proposed modelling is to address the main concerns inherent in concurrent systems including coordination and scalability. An actor-based language called Rebecca has been incorporated to model and analyze PobMC. Experiments using a Smart Mall System (SMALLS) case study show that the PobMC approach leads to effective policy-based management and is a feasible approach. Additionally, PobMC has the ability to enhance the existing approaches to support software adaptation. PobMC enables the coordination among system managers in order to adapt to system changes and avoid the potential conflicts thereby providing the main contribution of this research.
ABSTRAK

# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>CHAPTER</th>
<th>TITLE</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>DECLARATION</td>
<td>ii</td>
</tr>
<tr>
<td></td>
<td>DEDICATION</td>
<td>iii</td>
</tr>
<tr>
<td></td>
<td>ACKNOWLEDGEMENTS</td>
<td>iv</td>
</tr>
<tr>
<td></td>
<td>ABSTRACT</td>
<td>v</td>
</tr>
<tr>
<td></td>
<td>ABSTRAK</td>
<td>vi</td>
</tr>
<tr>
<td></td>
<td>TABLE OF CONTENTS</td>
<td>vii</td>
</tr>
<tr>
<td></td>
<td>LIST OF TABLES</td>
<td>xiii</td>
</tr>
<tr>
<td></td>
<td>LIST OF FIGURES</td>
<td>xiv</td>
</tr>
<tr>
<td></td>
<td>LIST OF ABBREVIATIONS</td>
<td>xvii</td>
</tr>
<tr>
<td></td>
<td>LIST OF APPENDICES</td>
<td>xix</td>
</tr>
<tr>
<td>1</td>
<td>INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>1.1</td>
<td>Background of the Problem</td>
<td>1</td>
</tr>
<tr>
<td>1.2</td>
<td>Statement of the Problem</td>
<td>3</td>
</tr>
<tr>
<td>1.3</td>
<td>Objectives of the Study</td>
<td>5</td>
</tr>
<tr>
<td>1.4</td>
<td>Scope of the Study</td>
<td>5</td>
</tr>
<tr>
<td>1.5</td>
<td>Significance of the Study</td>
<td>7</td>
</tr>
<tr>
<td>1.6</td>
<td>Structure of the Thesis</td>
<td>8</td>
</tr>
<tr>
<td>2</td>
<td>LITERATURE REVIEW</td>
<td>10</td>
</tr>
<tr>
<td>2.1</td>
<td>Definition of Policy</td>
<td>10</td>
</tr>
<tr>
<td>2.2</td>
<td>Policy Conflict</td>
<td>13</td>
</tr>
<tr>
<td>2.3</td>
<td>Policy Languages</td>
<td>15</td>
</tr>
<tr>
<td>2.3.1</td>
<td>Policy Description Language (PDL)</td>
<td>16</td>
</tr>
<tr>
<td>2.3.2</td>
<td>The Platform for Privacy Preferences Project (P3P)</td>
<td>16</td>
</tr>
</tbody>
</table>
2.3.3 The Web Services Policy Language (WSPL) 17
2.3.4 eXtensible Access Control Markup Language (XACML) 18
2.3.5 Policy Specification Language (Rei) 18
2.3.6 Web Service Policy Language (WS-Policy) 19
2.3.7 Policy and Domain Services Language (Kaos) 20
2.3.8 Ponder 21

2.4 Self Adaptive Software
2.4.1 Structural Adaptation 25
2.4.2 Behavioral Adaptation 25

2.5 Approaches to Develop Self Adaptive Software Systems 26
2.5.1 Knowledge-Based Approach 27
2.5.2 Policy-Based Approach 28

2.6 Comparative Evaluation 30
2.6.1 Outcome of the Comparative Evaluation 30
2.6.1.1 Flexibility 31
2.6.1.2 Scalability 32
2.6.1.3 Reliability 33
2.6.1.4 Performance 33
2.6.1.5 Maintainability 34
2.6.1.6 Interoperability 34

2.7 Coordination 36

2.8 Linear Temporal Logic (LTL) 39

2.9 A Systematic Survey on the Design of Self Adaptive Software System 41
2.9.1 Survey Results 46
2.9.2 General Statistics 46
2.9.3 Statistics Based on Self adaptive System Criteria 47
2.9.4 The Evaluation of Policy-Based Approaches 48
2.9.4.1 Evaluation Techniques 51
2.9.4.1.1 Dynamic Conflict analysis 51
2.9.4.1.2 Static Conflict Analysis 51
2.9.4.1.3 Policy Decoupling 51
2.9.4.1.4 Policy Classification 52
2.9.4.1.5 Policy Combination 52
2.9.4.1.6 Conflict Avoidance 52
2.9.4.1.7 Correctness of Adaptation 53
2.9.4.1.8 Check the System Scalability 53
2.9.4.2 Outcome of the Evaluation 53

2.10 Discussion 56
2.11 Summary 57

3 RESEARCH METHODOLOGY 58
3.1 Research Design 58
3.2 Research Procedure and Activities 60
   3.2.1 Literature Review 60
   3.2.2 Development of the System Structural Elements 60
   3.2.3 Specify the Conflict Avoidance Algorithms 61
   3.2.4 Evaluation 61
3.3 Operational Framework 62
3.4 Assumptions and Limitations 64
3.5 Case Studies 64
   3.5.1 A Smart Mall System SMALLS 67
   3.5.2 Smart House (SHOUSE) 67
   3.5.3 UTM Campus (UTMC) 68
   3.5.4 Objects of the Case Study 69

4 ANALYSIS OF THE CASE STUDY 70
4.1 The Illustrative Case Studies 70
   4.1.1 Policies 72
4.2 The Architecture of SMALLS System 76
4.3 The Implementation of the Case study 78
   4.3.1 Location-Based Services 78
   4.3.2 Personalised Location-based services 79
4.4 System Design and Architecture 81
   4.4.1 Evaluation Steps 84
   4.4.2 Distance Estimation 85
   4.4.2.1 The Location Tracking System 85
4.4.2.2 The Received Signal Strength Indicator (RSSI) 85
4.4.2.3 Distance Estimation using RSSI 87
4.4.3 Calculation of Path Loss 88
4.4.4 Location Determination in 2D 91
4.4.5 Location Determination in 3D 92
4.5 Policy management Approach 93
4.6 Specifying Adaptation System Specification 96
4.7 Summary 98

5 POLICY-BASED COORDINATION APPROACH (PobMC) 99

5.1 Overview of PobMC 99
  5.1.1 Policy Conflict Avoidance Processes 102
    5.1.1.1 Policy Refinement Process 103
    5.1.1.2 Dynamic Avoidance Process 107
  5.2 Evaluation of PobMC Algorithms 110
    5.2.1 Evaluation of Static Avoidance Algorithm 110
    5.2.2 Evaluation of Dynamic Avoidance Algorithm 114
  5.3 Modelling of PobMC 119
    5.3.1 Managers/Coordinators Layer 120
    5.3.2 Actors Layer 121
    5.3.3 Connectors Layer 121
  5.4 Responsibilities of Managers, Coordinators and Connectors 122
  5.5 Coordination Features 124
  5.6 Modelling PobMC Using Rebeca 126
  5.7 Operational Semantics 133
  5.8 Model-Checking PobMC Properties 137
  5.9 Specifying the Adaptation Behavior 144
    5.9.1 Syntax of LTL 144
    5.9.2 Specifying PobMC Adaptation Semantics 146
      5.9.2.1 Source/Target Behavior 147
      5.9.2.2 Controlling the Source Object Behavior 148
      5.9.2.3 Check and Avoid the Overlap Behavior 149
    5.9.3 Analysis of Adaptive SMALLS Services 151
      5.9.3.1 Safety 151
5.9.3.2 Liveness 153
5.9.3.3 Deadlock 156
5.10 Summary 157

6 POLICY CONFLICT ANALYSIS 159
6.1 Management Framework 159
6.2 Policy Conflict Classification 162
   6.2.1 Modality Conflicts 162
   6.2.2 Inconsistency 163
   6.2.3 Multiple Managers’ Conflicts 163
6.3 Overlap Analysis 163
   6.3.1 Overlap of Subjects 164
   6.3.2 Overlap of Roles 165
   6.3.3 Overlap of Targets 165
6.4 Conflict Analysis 166
   6.4.1 Modality Conflict Analysis 166
   6.4.2 Inconsistency Analysis 170
   6.4.3 Multiple Managers’ Conflicts Analysis 173
      6.4.3.1 Cycle 173
      6.4.3.2 Overlap 175
6.5 Summary 177

7 EVALUATION RESULTS AND DISCUSSION 179
7.1 Evaluation Results 179
   7.1.1 Results of PobMC Framework 179
   7.1.2 Results of Policy Conflicts Using Case Studies 181
      7.1.2.1 Results of Modality Conflict 183
      7.1.2.2 Results of Inconsistency Analysis 185
      7.1.2.3 Results of Cycle Detection 187
      7.1.2.4 Results of Overlap Detection 189
   7.2 Evaluation 191
      7.2.1 Evaluation of PobMC Modelling Results 191
      7.2.2 Evaluation of Policy Conflict Results 194
7.3 Summary 195
8 CONCLUSION 196
8.1 Research Summary and Achievements 196
8.2 Summary of the Main Contributions 199
8.3 Future Works 201

REFERENCES 203
Appendices A-D 225-240
# LIST OF TABLES

<table>
<thead>
<tr>
<th>TABLE NO.</th>
<th>TITLE</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Outcome of the Comparative Evaluation</td>
<td>35</td>
</tr>
<tr>
<td>2.2</td>
<td>Statistics Based on Publication Year</td>
<td>47</td>
</tr>
<tr>
<td>2.3</td>
<td>Taxonomy Based on Criteria</td>
<td>47</td>
</tr>
<tr>
<td>2.4</td>
<td>Evaluation Result of Policy-Based Approaches</td>
<td>55</td>
</tr>
<tr>
<td>3.1</td>
<td>Operational Framework</td>
<td>63</td>
</tr>
<tr>
<td>5.1</td>
<td>SMALLS Model Results Using Rebeca</td>
<td>130</td>
</tr>
<tr>
<td>7.1</td>
<td>Specification Properties Verification Results</td>
<td>181</td>
</tr>
<tr>
<td>7.2</td>
<td>Results of Modelity Conflict</td>
<td>184</td>
</tr>
<tr>
<td>7.3</td>
<td>Results of Inconsistency Analysis</td>
<td>185</td>
</tr>
<tr>
<td>7.4</td>
<td>Results of Cycle Detection</td>
<td>187</td>
</tr>
<tr>
<td>7.5</td>
<td>Results of Overlap Detection</td>
<td>190</td>
</tr>
<tr>
<td>7.6</td>
<td>Evaluation of Modelling</td>
<td>192</td>
</tr>
</tbody>
</table>
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>FIGURE NO.</th>
<th>TITLE</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Policy Configuration</td>
<td>12</td>
</tr>
<tr>
<td>2.2</td>
<td>Policy Conflict Classification</td>
<td>14</td>
</tr>
<tr>
<td>3.1</td>
<td>Research Design</td>
<td>59</td>
</tr>
<tr>
<td>4.1</td>
<td>Conceptual Class for Smart Mall System SMALLS</td>
<td>82</td>
</tr>
<tr>
<td>4.2</td>
<td>An Illustration of System Usage</td>
<td>83</td>
</tr>
<tr>
<td>4.3</td>
<td>Steps of the Case Study</td>
<td>84</td>
</tr>
<tr>
<td>4.4</td>
<td>The Curve-fitting Line</td>
<td>90</td>
</tr>
<tr>
<td>4.5</td>
<td>The Variation Curve</td>
<td>90</td>
</tr>
<tr>
<td>4.6</td>
<td>An Illustration of System Usage in 2D Area</td>
<td>91</td>
</tr>
<tr>
<td>4.7</td>
<td>An Illustration of System Usage in 3D Area</td>
<td>93</td>
</tr>
<tr>
<td>5.1</td>
<td>High Level View of PobMC System Architecture</td>
<td>101</td>
</tr>
<tr>
<td>5.2</td>
<td>Resolution of Conflicts Including the Constraints</td>
<td>104</td>
</tr>
<tr>
<td>5.3</td>
<td>Static Avoidance Algorithm</td>
<td>106</td>
</tr>
<tr>
<td>5.4</td>
<td>Dynamic Avoidance Algorithm</td>
<td>109</td>
</tr>
<tr>
<td>5.5</td>
<td>Deadlock Situation</td>
<td>111</td>
</tr>
<tr>
<td>5.6</td>
<td>Overlap Situation</td>
<td>113</td>
</tr>
<tr>
<td>5.7</td>
<td>System Throughput</td>
<td>115</td>
</tr>
</tbody>
</table>
5.8   The PobMC Model  
5.9   Higher Level Hierarchal Representation  
5.10  SMALLS Architecture Graph  
5.11  The Syntax of PobMC Model  
5.12  A typical PobMC Formal Model  
5.13  Transition System of Managers’ Activities in SMALLS  
5.14  An Abstraction of the Behavior of Source/Target  
5.15  An Abstraction of Controlling the Source Object Behavior  
5.16  An Abstraction of the Overlap Behavior  
5.17  A Finite Automaton for the Minimal Bad Prefixes of a Regular Safety Property  
5.18  An Example of a Deadlock Situation  
6.1   SMALLS Domains  
6.2   Policy Configuration  
6.3   Overlapping Target (T) Role (R) and Subject (S)  
6.4   Overlap of Subjects  
6.5   Overlap of Roles  
6.6   Overlap of Targets  
6.7   System Management  
6.8   Triple Overlap Between Two Policies  
6.9   Modality Conflict Algorithm  
6.10  Inconsistency Algorithm  
6.11  Cycle Detection Algorithm  
6.12  Multiple Managers’ Conflict Analysis Process
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.13</td>
<td>Multi Managers Algorithm</td>
<td>177</td>
</tr>
<tr>
<td>7.1</td>
<td>LTL Properties of SMALLS</td>
<td>180</td>
</tr>
<tr>
<td>7.2</td>
<td>Analysis Example of Conflict Avoidance</td>
<td>184</td>
</tr>
<tr>
<td>7.3</td>
<td>Performance Results for Modality Conflict Algorithm</td>
<td>181</td>
</tr>
<tr>
<td>7.4</td>
<td>Performance Results for Inconsistency Algorithm</td>
<td>186</td>
</tr>
<tr>
<td>7.5</td>
<td>Performance Results for Cycle Detection Algorithm</td>
<td>188</td>
</tr>
<tr>
<td>7.6</td>
<td>Performance Results for Overlap Algorithm</td>
<td>191</td>
</tr>
<tr>
<td>7.7</td>
<td>Performance Results for Overlap Algorithm</td>
<td>192</td>
</tr>
</tbody>
</table>
LIST OF ABBREVIATIONS

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADL</td>
<td>Architecture Definition Language</td>
</tr>
<tr>
<td>AP</td>
<td>Access Point</td>
</tr>
<tr>
<td>ECA</td>
<td>Event Condition Action</td>
</tr>
<tr>
<td>GPS</td>
<td>Global Positioning System</td>
</tr>
<tr>
<td>ISP</td>
<td>Internet Service Provider</td>
</tr>
<tr>
<td>IETF</td>
<td>Internet Engineering Task Force</td>
</tr>
<tr>
<td>KaOS</td>
<td>Policy and Domain Services</td>
</tr>
<tr>
<td>KB</td>
<td>A knowledge base</td>
</tr>
<tr>
<td>LOS</td>
<td>Line-Of-Sight</td>
</tr>
<tr>
<td>LTL</td>
<td>Linear Temporal Logic</td>
</tr>
<tr>
<td>MOP</td>
<td>Meta-Object Protocol</td>
</tr>
<tr>
<td>Mngr</td>
<td>Manager</td>
</tr>
<tr>
<td>NLOS</td>
<td>Non Line-Of-Sight</td>
</tr>
<tr>
<td>PDL</td>
<td>Policy Description Language</td>
</tr>
<tr>
<td>PDP</td>
<td>Policy Decision Point</td>
</tr>
<tr>
<td>PEPs</td>
<td>Policy Enforcement Points</td>
</tr>
<tr>
<td>PS</td>
<td>Policy Server</td>
</tr>
<tr>
<td>P3P</td>
<td>The Platform for Privacy Preferences Project</td>
</tr>
<tr>
<td>QoS</td>
<td>Quality of Service</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>----------------------------------------------</td>
</tr>
<tr>
<td>RSSI</td>
<td>Receive Signal Strength Indicator</td>
</tr>
<tr>
<td>Rebeca</td>
<td>REactive objECts lAnguage</td>
</tr>
<tr>
<td>SHOUSE</td>
<td>Smart House</td>
</tr>
<tr>
<td>SMALLS</td>
<td>Smart MALL System</td>
</tr>
<tr>
<td>TCP</td>
<td>Transmission Control Protocol</td>
</tr>
<tr>
<td>UTMC</td>
<td>Universiti Teknologi Malaysia Campus</td>
</tr>
<tr>
<td>XACML</td>
<td>eXtensible Access Control Markup Language</td>
</tr>
<tr>
<td>xADL</td>
<td>eXtensible Architecture Description Language</td>
</tr>
<tr>
<td>WSPL</td>
<td>Web Services Policy Language</td>
</tr>
<tr>
<td>2D</td>
<td>Two Dimensions</td>
</tr>
<tr>
<td>3D</td>
<td>Three Dimensions</td>
</tr>
</tbody>
</table>
# LIST OF APPENDICES

<table>
<thead>
<tr>
<th>APPENDIX NO</th>
<th>TITLE</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>List of Publications</td>
<td>225</td>
</tr>
<tr>
<td>B</td>
<td>Rebeca Code for SMALLS Using PobMC</td>
<td>227</td>
</tr>
<tr>
<td>C</td>
<td>Rebeca Language</td>
<td>230</td>
</tr>
<tr>
<td>D</td>
<td>List of Policies</td>
<td>236</td>
</tr>
</tbody>
</table>
CHAPTER 1

INTRODUCTION

This chapter introduces the current research with the background of the problem described first. After that, the problem statement, objective, scope, and importance of the study are described respectively.

1.1 Background of the Problem

The complexity of current software systems has led engineers to find innovative approaches for building, running, and managing software systems and services. In adaptive software, available information about environmental changes is used to simplify system complexity and improve behavior. Self-adaptation systems that are able to adjust runtime behavior have become a hot topic within the software engineering community. Although self-adaptation has been studied in a wide range of disciplines, only recently did the research community recognize its key role in enabling the development of future software systems that are able to self-adapt to changes that may occur in the system, its requirements, or the environment in which it is deployed (Cheng and Giese, 2009). However, the appropriate realization of the self-adaptation functionality remains a significant intellectual challenge and only recently have the first attempts in building self-adaptive systems emerged within specific application domains (Cheng et al., 2009). Policy-based management knowledge-based are some of the approaches recognized as an enabler of such a view.
In a large system, policies are often used as a means of management in a hierarchical fashion. A high-level policy guides a manager who may achieve goals by making lower-level policies that apply to other administrators lower in the hierarchy. However, the main challenge limits the development of such systems is the policy conflict. Conflicts may arise in the set of policies and during the refinement process between the high-level goals and the implementable policies (Lupu et al., 2007). The system must be able to handle conflicts including exceptions to normal authorization policies. For instance, in a large system there will be multiple human administrators specifying policies stored on distributed policy servers.

Policy conflicts can arise when multiple policies control the system behavior. A conflict occurs when an event triggers multiple actions that cannot occur together as specified by the system administrator. Human error is one obstacle to accurate access-control policies because the authors who assign and maintain these policies are prone to making specification errors that lead to incorrect policies. Access-control policies consist of a set of rules that dictate the conditions under which users will be allowed access to resources. These rules may conflict with each other. For example, an obligation policy may define an action that must be performed but there is no authorization policy to permit the manager to perform this activity. Conflict detection between management policies can be performed statically for a set of policies in a server as part of the policy specification process or at runtime (Michael et al., 1993; Sibley et al., 1993).

Obviously, there is a limitation in developing approaches that do not provide support for detecting and resolving conflicts. While a considerable attempt at static conflict detection has been presented (Lupu and Sloman, 1999), the very complex and crucial issue of dynamic conflict detection in a policy-based management has gone largely unresolved. Moreover, current research revealed that there is still a large class of policy conflict that cannot be determined statically. Static conflicts are used to detect specification errors and to reduce runtime conflicts that occur among rules whose event and condition parts can be statically matched. It may not be able to evaluate policy constraints because conflicts may depend on the runtime state of the system (Shiva et al., 2005).
1.2 Statement of the Problem

While policies have been successfully applied to manage various simple systems, the complexity and dynamism of active spaces pose several novel challenges. Though this thesis addresses these challenges in the context of self-adaptive software, the challenges and approach presented are applicable in the wider context of complex systems. This section presents the policy-related problems caused by the dynamism of systems.

In all systems, both subjects and target objects are represented as sets of objects called domains (Sloman and Twidle, 1994). The overlap relationship between domains is crucial to the discussion of policy conflicts as it is the contention of this work. There are several possibilities for overlap among policies corresponding to various combinations of overlap between objects in the subject and target object sets that are discussed in certain references (Lupu and Sloman, 1997; Lupu and Sloman, 1999). Overlap of some kind is of course the requisite for many kinds of relationships between policies that do not involve conflict.

According to the definition of conflicts (Dunlop et al., 2002; Charalambides et al., 2009; Davy et al., 2009), and the above-mentioned cases, to avoid the potential conflicts some information is necessary to define each type of conflicts (Khakpour et al., 2010). This section will introduce classification of various expected conflicts among system policies.

First, conflicts are expected when there is a triple overlap between the set of subjects' targets and actions of two or more policies with modality of opposite signs to the same subjects' actions and targets (Flegkas et al., 2009). For instance, “the subjects are authorized and forbidden to perform the action on the target objects.” Another example includes “the subjects are forbidden but required to perform the actions on the object.”

Secondly, contradiction occurs due to omission errors or conflicting requirements of the manager that enforces the policies (Lupu and Sloman, 1997; Lee,
2010). For instance, “an obligation policy defines an activity that must be performed but there is no authorization policy to perform the activity.” As an example, “the same subject cannot authorize the user and offer the resource.”

Thirdly, overlapping of domains (Sloman and Twidle, 1994; Simmons et al., 2006) relates to sharing of resources such as a gateway between two networks or a service between two or more applications. Overlapping leads to conflicts between policies when managers can be responsible for an object or that multiple policies apply to the object (Lupu and Sloman, 1999; Thebaut et al., 1999). In some situations, overlap is prevented by creating a new domain with an independent manager and all objects from the overlapping set are moved into this domain.

This work aims to provide an approach for building an adaptive-software system to support behavioral changes adaptation. Furthermore, this approach concentrates on the coordination between the system managers by developing a medium layer that includes some information about managers. The problem statement brings about the main research problem that is:

“How to enable the coordination between system managers to avoid the potential policy conflicts using appropriate technique in behavioral adaptation approach?”

To answer this question there is a set of related questions. Each constitutes a part or aspect of the main question that must be answered in order to find all the aspects of the proposed approach. These questions include:

(i) What are the important approaches to develop a self-adaptive system?
(ii) What are the main adaptive features to enable the proposed approach to be adapted to behavioral changes?
(iii) What does the management mean and include and what are the managed events in the proposed approach?
(iv) How could managers govern and manage policies to direct the system behavior?
(v) What are the requirements to make the proposed approach easy to use and development?
(vi) How to verify correctness of adaptation approaches in the context of a message communication application?

1.3 Objectives of the Study

The research objectives are mentioned based on the problem statement as follows:

(i) To develop and specify an adaptive architectural approach that may avoid the potential policy conflicts.
(ii) To specify the algorithm that supports the proposed architectural approach.
(iii) To evaluate the applicability of the proposed approach through application of the approach using a case study and conducting a synthetic environment experiment.

1.4 Scope of the Study

The study is divided into three research directions by researches in adaptation policy-based management and coordination. The above directions have been considered as the scope of the research.

Firstly, the term adaptation in the software context refers to changes that happen during its lifetime involving “planned activities” and “unplanned phenomena”(Reiss, 2005). The systems are considered high assurance since errors during execution could result in high cost if adaptation is not carefully planned. Furthermore, systems must continue to evolve to correct defects, additions, or removals of functional behaviors and adaptation to the operating environment. Ideally, adaptation should occur without the need to interrupt system operations.
Secondly, current approaches have been acknowledged most often as a methodology that provides adaptability. Therefore, this coordination approach (Wang et al., 2005) makes easy adaptation of new rules and system resource changes, thus enabling dynamic system management according to the alteration of system rules. Additionally, user experiences and expertise, as well as other heuristics can be encoded into policies to help or guide resource management. Furthermore, it is one of the key approaches that is recognized as an enabler or connector of self-management properties for telecommunications networks (Loyola, 2007). Employing policies as a paradigm to modify self-adaptive systems was given considerable attention recently.

Thirdly, coordination is the harmonious adjustment or interaction of different things to achieve a goal or effect (Canal et al., 2005). Coordination languages and models are being developed to address the problem of managing the interactions among concurrent and distributed processes (De Vries et al., 2009). The underlying principle is the separation of computations by components and their interactions (Colman and Han, 2005; Gesbert et al., 2007). To achieve correct coordination (Zhu, 2008), rather than only considering dependency relations between multiple adaptations, this approach further focuses on dependency relations between managers at runtime. This work considers a number of features including specification to identify and measure achievement of managerial goals to insure that the modelling provides mechanisms for structuring or modularising coordination activities and to verify that coordinated managers do not have any explicit action that may affect the coordination (Nogueira et al., 2012; Wang et al., 2012).

1.5 Significance of the Study

Currently, software systems increasingly rely on dynamically adaptive programs to respond to changes in their physical environment. Examples include ecosystem monitoring and disaster relief systems. These systems are considered high assurance because errors during execution could result in injury, loss of life, environmental impact, and financial loss. Furthermore, systems must continue to
evolve to correct defects, addition, or removal of functional behaviors, and adaptation to the operating environment. Ideally, the evolution should occur without the need to interrupt system operation (Wang et al., 2006). However, evolution occurs with high cost if it is not carefully planned.

Static and dynamic conflicts were considered as two distinct classes that need to be understood and independently managed (Dunlop et al., 2002). The distinction between these two classes is important because detecting and resolving conflict can be computationally intensive, time consuming, costly, and is most preferably done at compile-time. However, dynamic conflict is quite unpredictable because it may or may not proceed to a state of realized conflict. This class of conflict must be detected at runtime.

The considerable attempt at static conflict detection, the very complex and crucial issue of dynamic conflict detection in a policy-based management, has gone largely unresolved. Moreover, current research has revealed a large class of policy conflict that cannot be determined statically.

Existing works in developing adaptive systems have several challenges that must be addressed appropriately. The most fundamental challenge is to verify whether the system is operating as required (Khakpour et al., 2010). Furthermore, existing policy-based systems do not provide any techniques to uncover such conflicts. The initiated rules are assumed to execute without any failures because there are no mechanisms to monitor and verify rule enforcement. According to the knowledge of these authors, existing systems do not reason about concurrent rule enforcement, define enforcement ordering, or investigate the effects of different policies.

Consequently, providing a guideline to avoid policy conflicts between different domain configurations and resolve the conflicts during implementation time is very important in order to simplify software evolution tasks and minimize evolution efforts.
1.6 Structure of the Thesis

This Thesis is composed by eight Chapters and four Appendixes. Chapter 1 introduces the Thesis providing a general outline of policy-based management and the policy conflict problem. We also state the motivations of the Thesis and its main contributions to the state of the art.

**Chapter 2:** presents some background material that constitutes the foundations of the Thesis. Concretely, we present some crucial material on policy, self adaptive software. Moreover, we present a comparative evaluation on the approaches to develop self adaptive system. In this chapter, we provide a systematic literature review on self adaptive approaches. Finally, we conclude the weaknesses of the previous approaches.

**Chapter 3:** Provides the research methodology that describes research design and formulation of research problems and validation considerations. This chapter describes research design and procedure that is utilized in this research work. It also describes the case studies that are applied to evaluate PobMC as well as describes the evaluation process that is conducted in this research. Moreover, it explains some research assumptions and limitations.

**Chapter 4:** This Chapter provides analysis of the case study. Some examples are provided in the analysis to illustrate the development of the proposed approach. Some policy examples are provided clarifying the management of the system. The adaptability of the case studies is using the criteria in chapter 2 is discussed in this chapter.

**Chapter 5:** In this chapter the system architecture of the proposed approach is presented in details, the evaluation of the algorithm is discussed. Furthermore, the modelling of PobMC is provided using the actor based language Rebeca. This chapter also specifies the adaptation behaviour of PobMC approach.
Chapter 6: This chapter presents static analyses to address the inconsistencies; multiple managers’ conflicts and modality conflict, when there are two or more policies are enforced simultaneously. Moreover, the chapter presents a classification of policy conflicts, and then provides temporal patterns to avoid each class of policy conflicts, and to ensure that policies are enforced correctly.

Chapter 7: This chapter describes the evaluation results of PobMC as well as its relevant discussions. It contains the results of the case study and the experiment. Then, each result is furnished with sufficient discussions about the specific condition found in each case. Finally, results of the investigative evaluation are described.

The Summary Conclusions of the thesis are provided in Chapter 8. We review the contributions of the Thesis, provide additional discussion of relevant issues of the Thesis and provide some directions for future work.

Appendix A references the author’s publications related to this Thesis. Appendix B includes the modelling of case studies using Rebeca language. Appendix C presents the actor based language Rebeca. Finally, Appendix D includes the list of policies that used in the implementation of the proposed approach in this thesis.
REFERENCES


Colman, A. and Han, J. (2005). Coordination systems in role-based adaptive software. 7th International Conference on Coordination Models and


generation. Morgan Kaufmann.

Software Components for Mobility Computing, In: EDOC 2006: 10th IEEE
International Enterprise Distributed Object Computing Conference,
Washington, DC, USA.,

Talcott, C. L. (2006). "Coordination models based on a formal model of distributed
object reflection." Electronic Notes in Theoretical Computer Science 150(1):
143-157.

Thebaut, S., Scott, W., Rustici, E., Kaikini, P., Lewis, L., Malik, R., and Wohlers, T.
Trademark Office.

for Software Components: Case-Based Reasoning Approach. In Computer
Software and Applications Conference, 2009. COMPSAC'09. 33rd Annual

and reasoning: A comparison of KAoS, Rei, and Ponder." The Semantic
Web-ISWC 419-437.

32nd Conference on Current Trends in Theory and Practice of Computer

services. In W3C Workshop on Constraints and Capabilities for Web
Services (October 2004).

Building adaptive systems using Ensemble. Cornell University.


edu/Faculty/Vouk/vouk_se. html.


